Modeling, simulation, Monte Carlo methods

- **Modeling and simulation of natural processes**
  - ➢ **introduction and general concepts**
- **Monte Carlo methods**

Follow the course/slides from
B. Chopard et al., coursera lectures, University of Geneva
S. Paltani, Statistical Course for Astrophysicits, University of Geneva

Prof. dr hab. Elżbieta Richter-Wąs

Examples of natural processes

- Physics (Fluid mechanics), astrophysics, chemistry, climatology,...
- Environmental sciences (river modeling, Volcano plume)
- Biology: (Tissue growth), pattern on animal skins, cells, organs
- Ecosystems: competition between species, ant behavior, equilibrium between forest and savanna, propagation of epidemia,...
- Finance, social sciences, traffic, pedestrian motion,..
- ...

What is a model?

- This is not an apple just its graphical representation

What is a model?

Many possible definitions:

- Simplified abstraction of reality, allowing us to better describe and understand it
- An abstraction in which only the essential ingredients are retained, according to the question we ask about the system.
- It is the respresentation of a phenomena in a mathematical or computer-based language,

Computational Science

Many skills are needed to build a new model, to run it and analyze its results.

- Computational Science is an emerging, multidisciplinary domain, based on the idea of **"computational thinking"**.
- A computer-based description offers a new language, a new methodology to address scientific challenges, far beyond the scope of traditional numerical methods, and in fields where these classical approaches hardly apply.

Computational Science

- **Modeling and simulation** is a central part of computational sciences. It is a response to the new questions scientists want to solve, resulting from the avalanche of new experimental data, and the need to integrate many processes together rather than specializing in one single problem.
- A computational scientists needs to be a physisict, a mathematician, a computer scientist, a biologist, an economists...

Why a model? What is a good model?

Why a model?

- Describe, classify, but mostly
- Understand
- Predict
- Control a phenomena

What is a good model?

It depends on the question. Several models may be necessary for studying different aspects of the same phenomena

> *Everything should be made as simple as possible but not simpler*

A Einstein

Level of reality

The same system can be described at different scales, and different methods apply depending on the scale one is interested in:

- atoms, molecules, fluid elements, pressure field, climat
- cells, tissues, organs, living beings
- mechanical parts, cars, traffic

- One has to identify the important ingredients and their mutual interactions.
- Often, one defined a model at finer scale than the scale at which we ask a question.

Several models/ different language of description

Partial differential equation for a fluid;

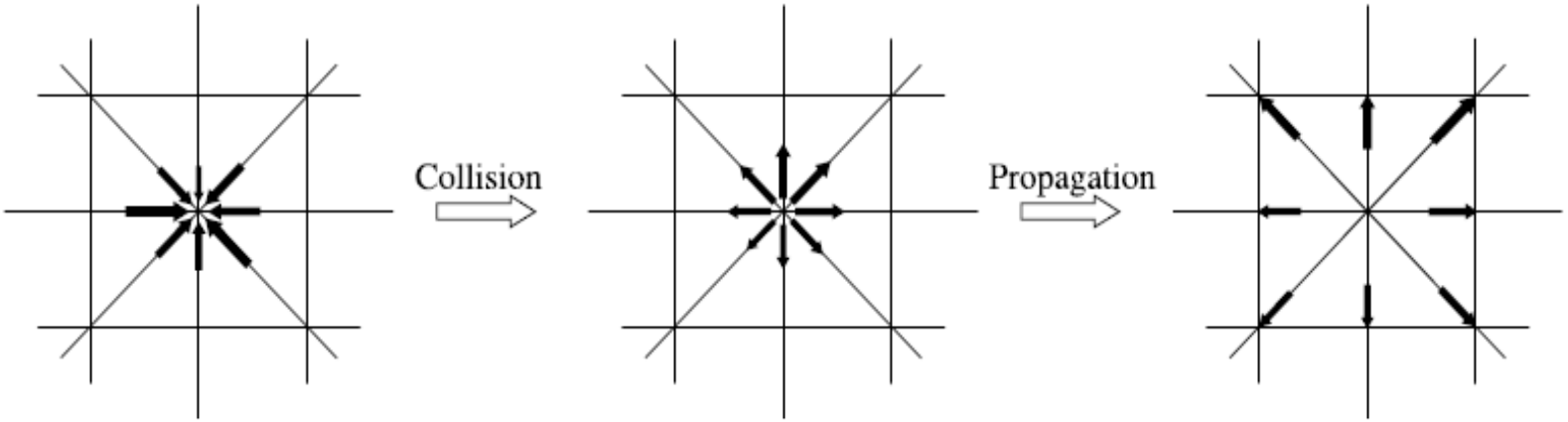
$$\partial_t \mathbf{u} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\frac{1}{\rho}\nabla p + \nu \nabla^2 \mathbf{u}$$

phenomena → PDE→ discretisation → numerical solution

…to a virtual model of reality

One considers a discrete universe as an abstraction of the real word

phenomena → computer model



- Mesoscopic *Rule* describing the phenomena

Example of modeling method

- N-body systems, molecular dynamics
- Mathematical equations, ODE, PDE
- Monte-Carlo methods (equilibrium, dynamic, kinetic)
- Cellular Automata and Lattice Boltzmann methods
- Multi-agents systems
- Discrete Events simulation
- Complex networks

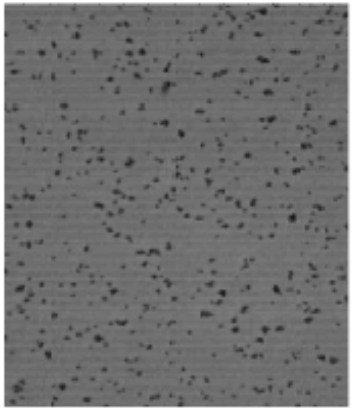
From a model to a simulation

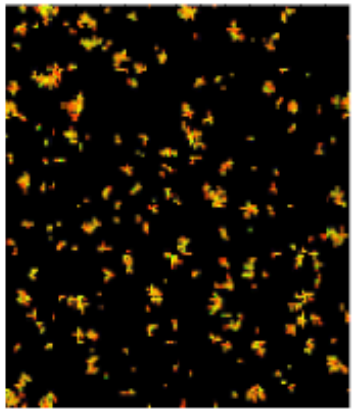
- Once a model is specified, one need to program it, to run it (many times) and to study the results.
- It is a numerical experiment in **computer based virtual universe**
- One need to understand computer programs, software engineering, algorithms, data-structures, hardware (parallel machines, GPUs), code optimization, data-analysis.

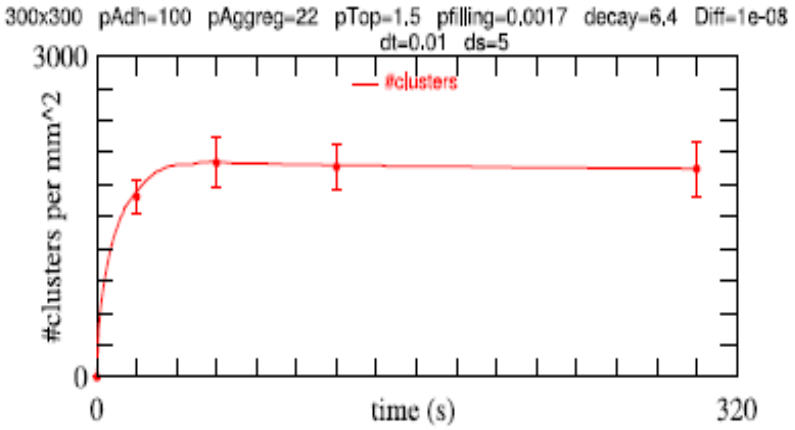
From a model to a simulation

- The program needs to be verified (did we really implemented the model?)
- The model should be validated (run benchmarks with known results).
- One need enough knowledge of the phenomena to judge if its predictions are acceptable in new situations.

From a model to a simulation: illustration







Space and time

- Natural processes occurs in space and evolve over time (spatially extended dynamical systems).
- For instance the atmospheric temperature is different from one place to another, and changes over time.
- Also, a car on a road changes position as time goes on.
- Sometime one is only interested in the time evolution of a quantity, regardless of the spatial location (e.g the number of individuals) in a population
- Sometime, a process is stationary (no time evolution). Then only the spatial variations are of interest (e.g temperature in a room, in the middle or near the windows).

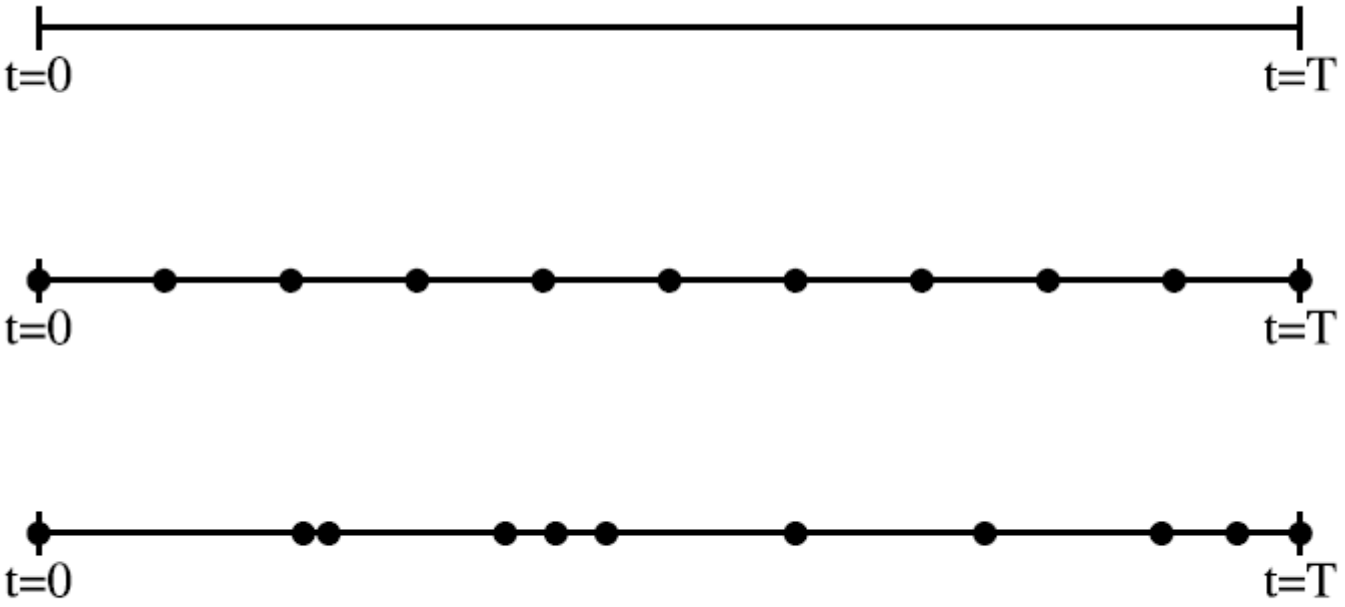
Time evolution

- To capture the temporal dimensions in a model, there are several ways:
- Time takes any real values (as physics suggests). Only mathematical models can deal with this approach (differential equations)
- Otherwise, the duration of the process is broken up in small time intervals Δt and one describes the state of the system at each of these **time-step** $t_0 = 0, t_1 = \Delta t, \ldots, t_n = n\Delta t \ldots$.
- The time is discretized, but the process is followed **continuously** over its duration.

Time evolution

- Alternatively, we can only focus on the interesting moments of a process
- In a queue in front of a post office booth, one can simply consider the time at which a remarkable event occurs. For instance a new customer enters, or a previous one is done.
- The time t at which an event occurs can be any real value.
- The time is not discretized but the evolution of the system is broken up according to events.
- This is the so-called Discrete-Event-Simulation (DES) approach

Time evolution



Modeling space: Eurelian approach

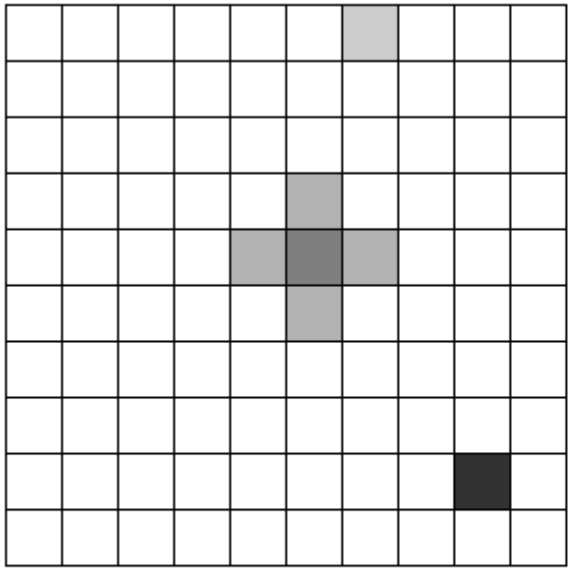
To include the spatial dimensions in a model, there are also different ways.

- One can take the point of view of an observer who sits at a fixed position $\vec{x}$ in space and records what he sees.
- For instance the local atmospheric pressure $p(\vec{x}, t)$.
- Or the number of cars that passed by every minute.
- This is the so-called **Eulerian** approach: attach a property of the system at each spatial locations.
- Space can be continuous (mathematical models) or discretized in cells, forming a mesh covering the region of interest.

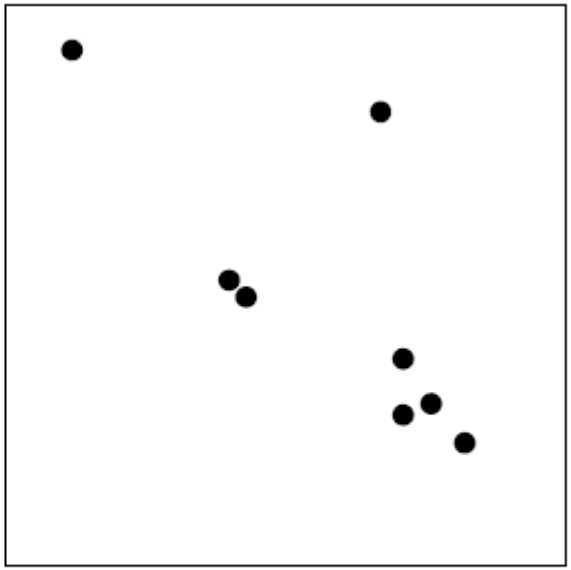
Modeling space: Lagrangian approach

- Alternatively, one can give the position of all the objects of interest, as a function of time.
- For instance the movement of the Moon is described by its trajectory $\vec{x}(t)$, where $\vec{x}$ is a continuous variable.
- In a traffic model, one can give the positions over time of all the cars.
- This is the so-called **Lagrangian** approach: the observer take the point of view of the moving objects.

Modeling space



Eulerian point of view



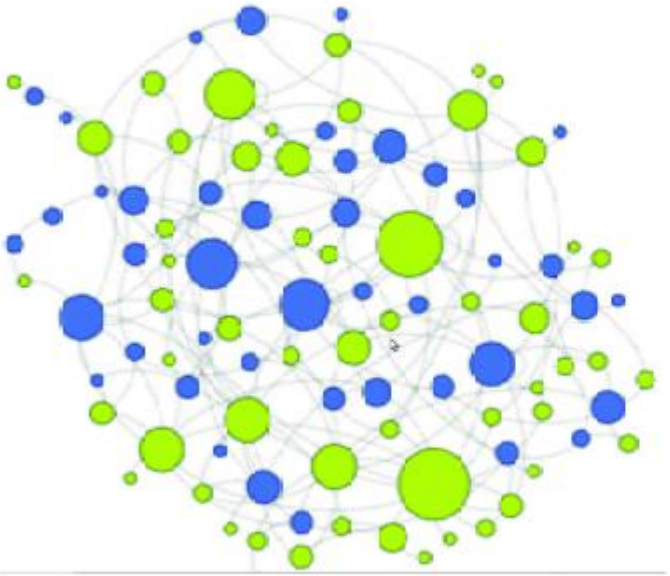
Lagrangian point of view

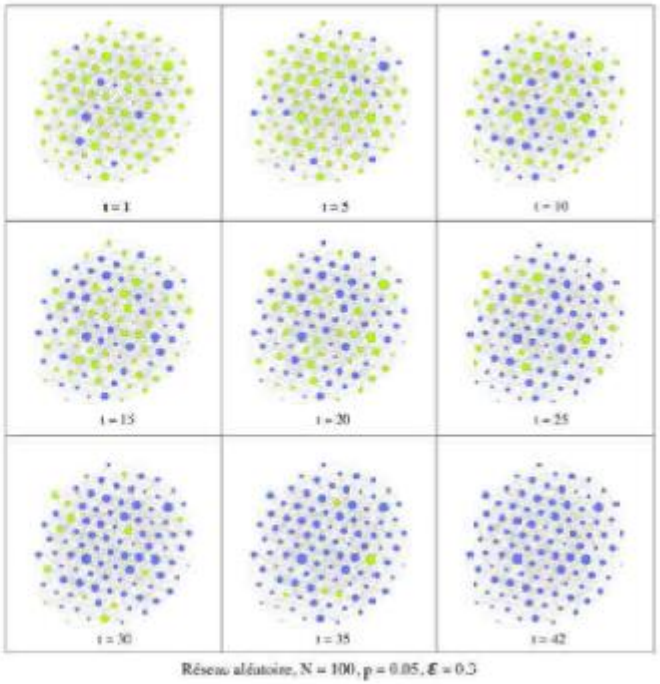
Beyond the physical space: complex networks

- In many systems, it is not so much the exact spatial positions of the components of a system that matters
- It is rather whether these components see each others, or can interact.
- This is typically the case in social systems. Two persons can be very far away but still interact a lot by phone or other means
- For instance, the agent in an economical model can be represented as a **graph** (or a complex network): an edge connects pairs of agents that exchange information, money, goods,..
- Obviously, such a graph can be dynamical: creation of new links or destruction of old ones.

Example

A model of opinion propagation in a social network





Réseau aléatoire, $N = 100$, $p = 0.05$, $\mathcal{E} = 0.3$

(Lino Velasquez, UNIGE)

Complex networks

- Dynamical systems on complex networks is a fast developing field
- Graph topology imposes a rich "spatial" structure which constrains the dynamics
- Many quantities characterize the graph topology and can be related to some global properties of the system: degree distribution, clustering coefficient, centrality measures, assortativity, etc.

Monte Carlo methods

- The goal of Monte-Carlo methods is the sampling of a process in order to determine some statistical properties
- For instance, we toss a coin 4 times. What is the probability to obtain 3 tail and 1 head?
- Mathematics gives us the solution:

$$P(3 \text{ head}) = \left(\begin{array}{c} 4 \\ 3 \end{array} \right) \left(\frac{1}{2}\right)^3 \left(1 - \frac{1}{2}\right)^1 = \frac{1}{4}$$

- But we could also do a simulation

A Monte Carlo computer simulation

```
from random import randint

success=0

attempts=10000
for i in range(attempts):
    if randint(0,1)+randint(0,1)+randint(0,1)+randint(0,1)==3:
        success+=1

print "Number of attempts=", attempts
print "Number of success=", success
```

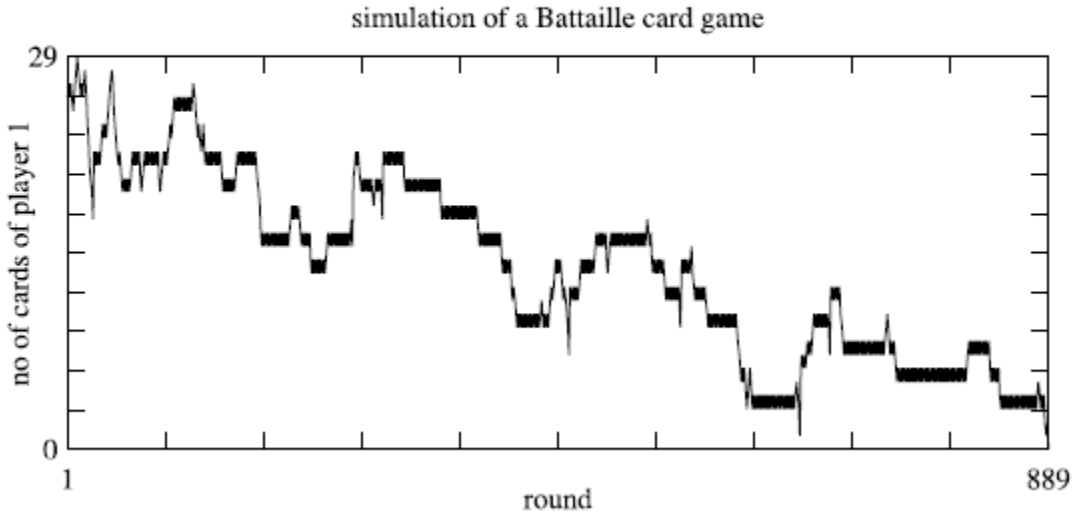
We get for instance:

```
Number of attempts= 10000
Number of success= 2559
```

A more difficult problem

- For the coin tossing problem, no need for a simulation
- But we can think of other problems for which probability theory could hardly be applied
- For instance: what is the average duration of the card game called "war" (or battle)?

The *war* card game with 52 cards



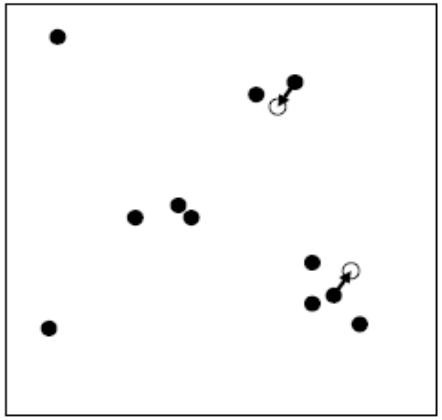
Historical note

- The method was name in the 1940s by **John von Neumann**, **Stanislaw Ulam** and **Nicholas Metropolis** after the name of the *Monte-Carlo casino*, where Ulam's uncle used to gamble ...and loose his money
- The motivation was to find out the probability that a Canfield solitaire will finish sucessfully.
- Ulam found it easier to play many Canfield solitaires and estimate the number of successes, rather than trying to apply combinatorics and probability theory.
- Then the Monte-Carlo methods was successfully applied to the *Manhattan project* (nuclear weapon) in the Los Alamos National Laboratory.

Markov-chain Monte Carlo (MCMC)

- We consider a stochastic process whose goal is to explore the state space of a system of interest.
- Let x be a point in this state space. Let us asssume that this point moves across the space by jumping randomly to another point x'.
- The jump from location x to location x' takes place with probability $W_{x \to x'}$. This advanced the system time from t to $t+1$ (Markov chain)

- We want this process to sample a prescribed probability $\rho(t, x)$. This stochastic process should be at point x at time t with a probability $\rho(t, x)$.
- How do we choose $W_{x \to x'}$?



$$\rho \propto \exp(-E(x)/k_B T)$$

Sampling the diffusion equation in 1D

The probability that our random exportation is at location x at time t is

$$p(t+1,x) = \sum_{x'} p(t,x') W_{x' \to x}$$

- Let us consider a 1D discrete space: $x \in \mathbf{Z}$.
- where one can move to the right with probability W_+, to the left with probability W_- and stay still with probability W_0.
- The equation for $p(t,x)$ simplifies to

$$p(t+1,x) = p(t,x-1)W_+ + p(t,x)W_0 + p(t,x+1)W_-$$

Diffusion equation in 1D

- The diffusion equation is $\partial_t \rho = D \partial_x^2 \rho$
- Which can be discretized as

$$\rho(t + \Delta t, x) = \rho(t, x) + \frac{\Delta t D}{\Delta x^2} \left(\rho(t, x - 1) - 2\rho(t, x) + \rho(t, x + 1)\right)$$

- to be compared with

$$p(t + 1, x) = p(t, x - 1)W_+ + p(t, x)W_0 + p(t, x + 1)W_-$$

- In order to have $p = \rho$, one need $W_+ = W_- = \Delta t D/(\Delta x)^2$ and $W_0 = 1 - 2\Delta t D/(\Delta x)^2 = 1 - W_+ - W_-$, and thus $\Delta t D/(\Delta x)^2 \leq 1/2$

- Therefore a random walk is a way to sample a density ρ that obeys the diffusion equation.
- With a random walk, it is easy to add obstacles, or aggregation processes, hard to include in the differential equation.

More general case: Master equation

The probability to find the random exploration at location x at time t is $p(t,x)$ given by

$$
\begin{aligned}
p(t+1,x) &= \sum_{x'} p(t,x')W_{x'\to x} \\
&= \sum_{x'\neq x} p(t,x')W_{x'\to x} + p(t,x)W_{x\to x} \\
&= \sum_{x'\neq x} p(t,x')W_{x'\to x} + p(t,x)(1-\sum_{x'\neq x} W_{x\to x'}) \\
&= p(t,x) + \sum_{x'\neq x} [p(t,x')W_{x'\to x} - p(t,x)W_{x\to x'}]
\end{aligned}
$$

Detailed balance

In a steady state, the condition $p(x) = \rho(x)$ requires that

$$\sum_{x' \neq x} [\rho(x')W_{x' \to x} - \rho(x)W_{x \to x'}] = 0$$

We can then choose $W_{x \to x'}$ according to the **detailed balance** condition

$$\rho(x')W_{x' \to x} - \rho(x)W_{x \to x'} = 0$$

Metropolis Rule

Let us consider a physical system at equilibrium whose probability to be in state x is given by the Maxwell-Boltzmann distribution

$$\rho(x) = \Gamma \exp(-E(x)/kT)$$

We can sample this distribution with a stochastic process by choosing $W_{x \to x'}$ according to the **Metropolis rule**:

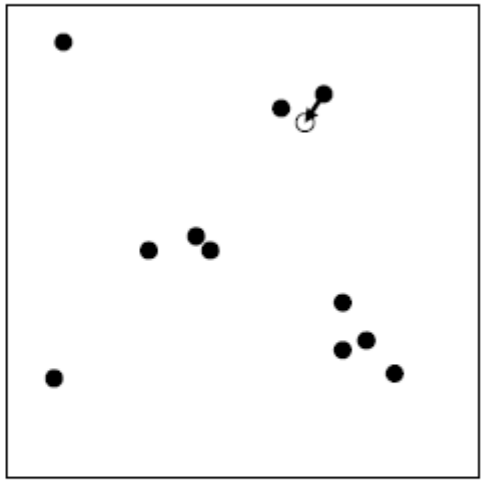
$$W_{x \to x'} = \begin{cases} 1 & \text{si } E' < E \\ \exp[-(E' - E)/kT] & \text{si } E' > E \end{cases}$$

Metropolis Rule in practice

- In a gas, one selects one particle at random.
- One moves it by an amount Δx.
- One computes the energy E' of the gas with this new position.
- One accepts this change if

$$\text{rand}(0,1) < \min(1, \exp[-(E' - E)/kT])$$

- By sampling ρ with $W_{x \to x'}$, one can compute average physical properties, such as for instance the pressure in the gas.



Metropolis Rule in practice

The Metropolis obeys the detailed balance

Let us assume that $E' > E$. Detailed balance is obeyed because

$$\begin{aligned} \rho(x)W_{x\to x'} &= \Gamma \exp(-E/kT)\exp[-(E'-E)/kT] \\ &= \Gamma \exp(-E'/kT) \\ &= \rho(x') \times 1 \\ &= \rho(x')W_{x'\to x} \end{aligned}$$

And similarly if $E' \leq E$

Glauber Rule

This is an alternative to the Metropolis rule. $W_{x\to x'}$ is given by

$$W_{x\to x'} = \frac{\rho(x')}{\rho(x) + \rho(x')}$$

which also clearly obeys detailed balance
With $\rho = \Gamma \exp(-E(x)/kT)$, one obtains

$$W_{x\to x'} = \frac{\exp(-E'/kT))}{\exp(-E/kT) + \exp(-E'/kT)}$$

Kinetic/Dynamic Mote Carlo

Let us consider the chemical equations

$$A \xrightarrow{k_1} B \qquad B \xrightarrow{k_2} A$$

They can be written as an ordinary equation

$$\frac{d}{dt}\begin{pmatrix} A \\ B \end{pmatrix} = \begin{pmatrix} -k_1 & k_2 \\ k_1 & -k_2 \end{pmatrix}\begin{pmatrix} A \\ B \end{pmatrix}$$

Analytical solution

$$A(t) = \frac{k_2}{k_1+k_2}(A_0+B_0) + \frac{A_0k_1 - B_0k_2}{k_1+k_2}e^{-(k_1+k_2)t}$$

$$B(t) = \frac{k_1}{k_1+k_2}(A_0+B_0) - \frac{A_0k_1 - B_0k_2}{k_1+k_2}e^{-(k_1+k_2)t}$$

where A_0 and B_0 are the initial concentration of A and B.
When $t \to \infty$,

$$A \to A_\infty = \frac{k_2}{k_1+k_2}(A_0+B_0) \qquad B \to B_\infty = \frac{k_1}{k_1+k_2}(A_0+B_0)$$

Monte Carlo simulation

1 One defines a time step Δt, small enough so that $k_1 \Delta t$ et $k_2 \Delta t$ are smaller than 1. They are the **probabilities** that, during Δt, one A particle get transformed into one B particle, or conversely.

2 One chooses randomly a particle among the $N = A(t) + B(t) = \text{const}$ of them. (In practice one chooses a A particle $\text{rand}(0,1) < A/(A+B)$, and a B particle otherwise.

3a If a A particle was chosen, it is transformed into a B particle provided $\text{rand}(0,1) < k_1 \Delta t$. Then $A = A - 1$, $B = B + 1$.

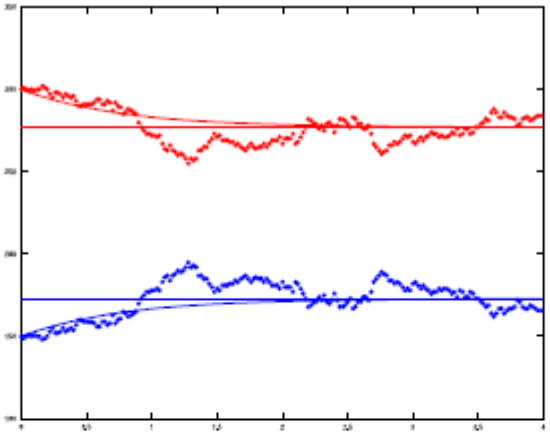
3b If a B particle was chosen, it is transformed into a A particle, provided $\text{rand}(0,1) < k_2 \Delta t$. Then $A = A + 1$, $B = B - 1$.

4 (2) and (3) are repeated N times and the physical time t is incremented by Δt: $t = t + \Delta t$

5 One repeats (2)-(4) until $t = t_{max}$

Monte Carlo simulation

Results



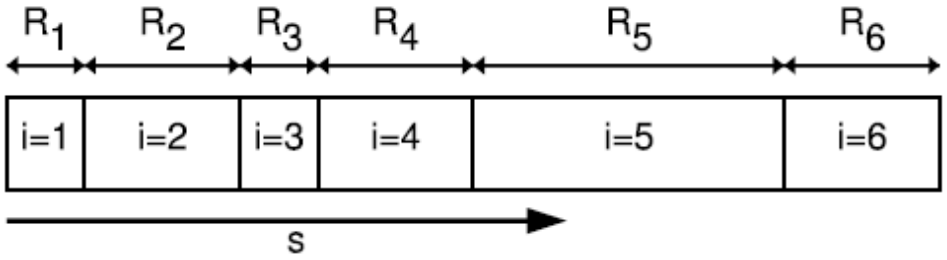
$\Delta t = 0.02$ and $k_1 = 0.5$, $k_2 = 0.8$.

The Monte-Carlo simulation fluctuate around analytic solution.

We should average over several runs

Gillespie's algorithm

- Let r_i be the rate at which the possible events occur in the system. $i = 1, \ldots n$.
- For instance, $r_i = kAB$ for a reaction $A + B \to C$
- Let $R_i = \sum_{j=1}^{i} r_i$ be the cumulative rates.
- Choose one event k at random by picking a random number $s = \text{rand}(0, 1)$: One chooses the k which verifies $R_{k-1} < sR_n < R_{k+1}$ (probability proportional to rate).



- Execute the selected event (for instance a molecule A and molecule B disappear to produce a new molecule C).
- Advance time as $\Delta t = \ln(1/s')R_n^{-1}$, with $s' = \text{rand}(0, 1)$.

More on Monte Carlo methods

- have been invented in the context of the development of the atomic bomb in the 1940's
- are a class of computational algorithms
- can be applied to vast ranges of problems
- are not a statistical tool
- rely on repeated random sampling
- provide generally approximate solutions
- are used in cases where analytical or numerical solutions don't exist or are too difficult to implement
- can be used by the Lazy Scientist^{TM} even when an analytical or numerical solution can be implemented

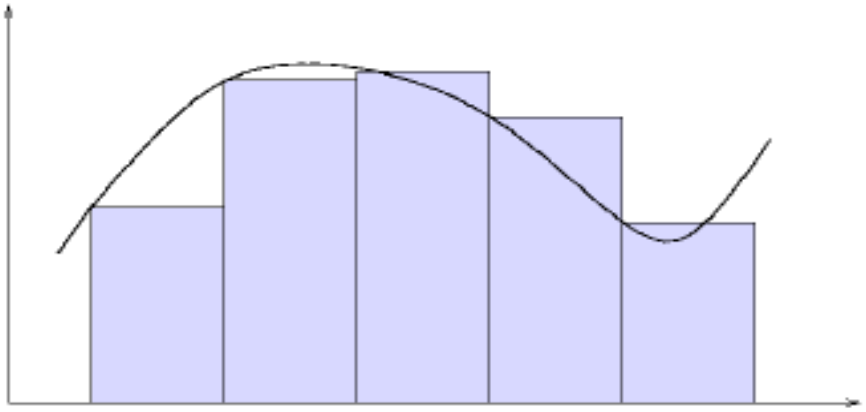
Monte Carlo methods

Monte-Carlo methods generally follow the following steps:

1. Determine the statistical properties of possible inputs
2. Generate many sets of possible inputs which follows the above properties
3. Perform a deterministic calculation with these sets
4. Analyze statistically the results

The error on the results typically decreases as $1/\sqrt{N}$

Numerical integration



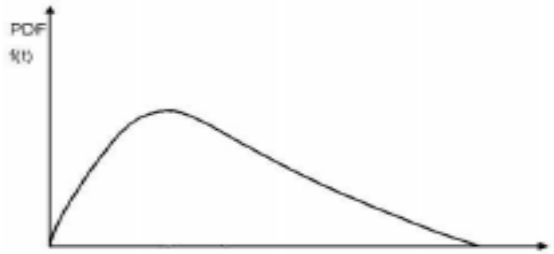
Most problems can be solved by integration

Monte-Carlo integration is the most common application of Monte-Carlo methods

Basic idea: Do not use a fixed grid, but random points, because:

1. Curse of dimensionality: a fixed grid in D dimensions requires N^D points
2. The step size must be chosen first

Error estimation

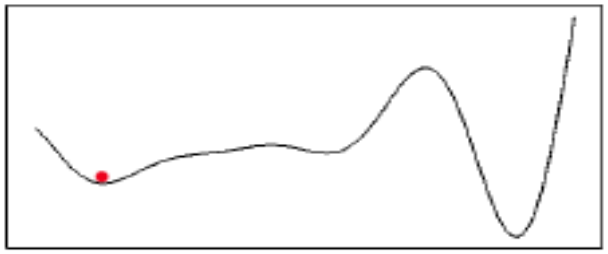


Given any arbitrary probability distribution and provided one is able to sample properly the distribution with a random variable (i.e., $x \sim f(x)$), Monte-Carlo simulations can be used to:

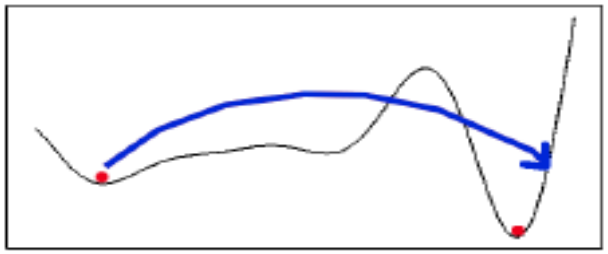
- determine the distribution properties (mean, variance,...)
- determine confidence intervals, i.e. $P(x > \alpha) = \int_{\alpha}^{\infty} f(x)\mathrm{d}x$
- determine composition of distributions, i.e. given $P(x)$, find $P(h(x))$, $h(x) = x^2; cos(x) - sin(x); \ldots$

Note that these are all integrals!

Optimisation problems

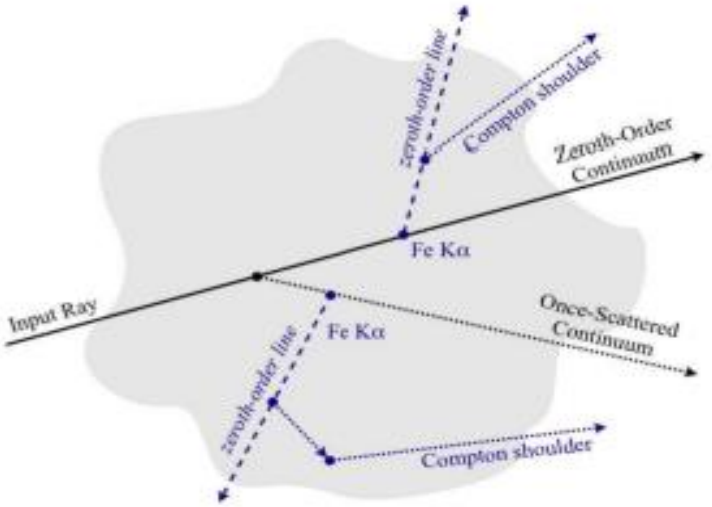


Numerical solutions to optimization problems incur the risk of getting stuck in local minima.



Monte-Carlo approach can alleviate the problem by permitting random exit from the local minimum and find another, hopefully better minimum

Numerical simulations

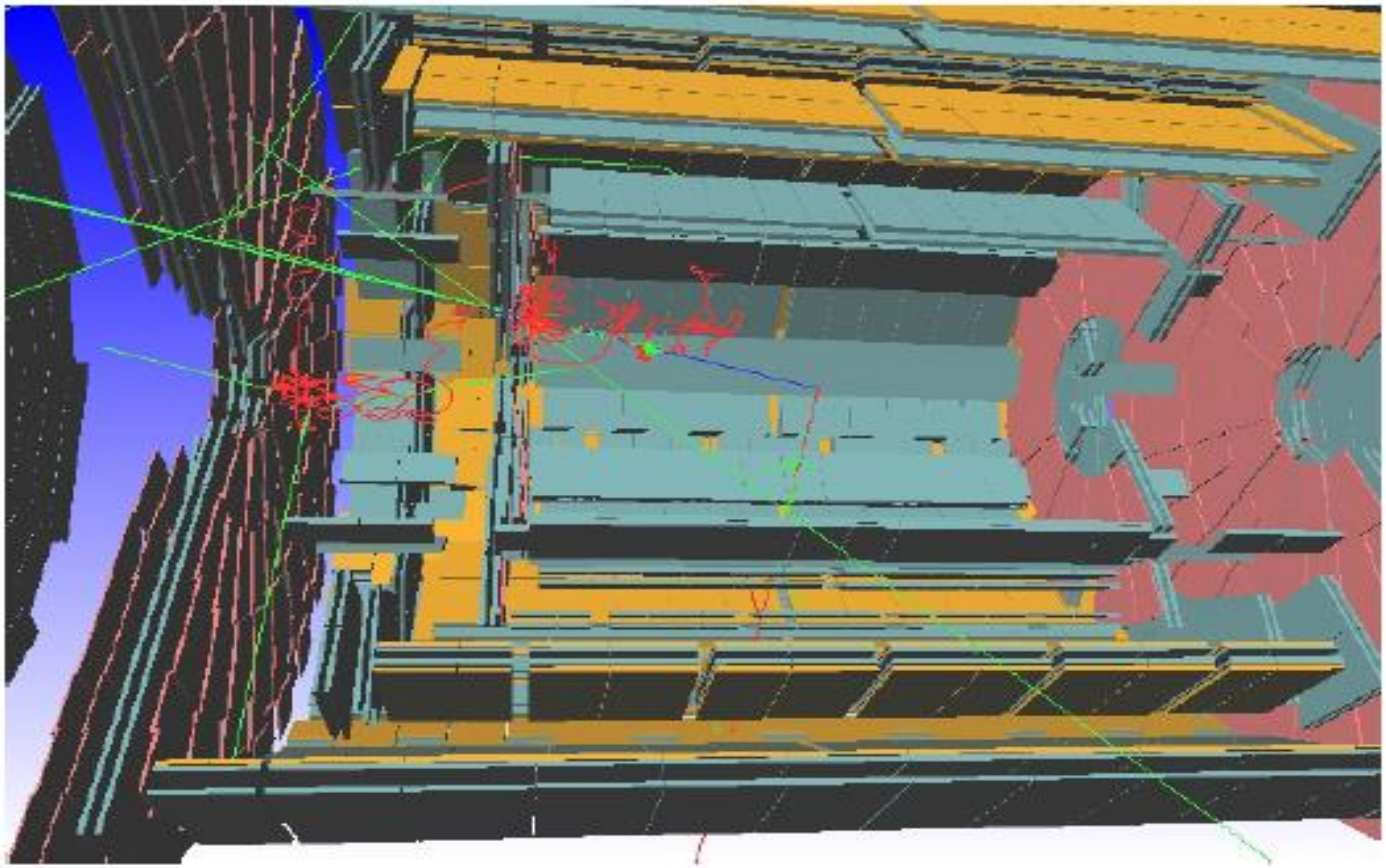


- Radiation transfer is Google-wise the main astrophysical application of Monte-Carlo simulations in astrophysics
- In particle physics and high-energy astrophysics, many more physical processes can be simulated

Some physical processes are discretized and random by nature, so Monte-Carlo is particularly adapted

Numerical simulations

GEANT4



GEANT4 is also used to determine the performance of X-ray and gamma-ray detectors for astrophysics

Example: probability estimation

Head vs tail probability

What is the probability to obtain either 3, 6 or 9 heads if one draws a coin ten times?

- Binomial probability:
  $P = B(3; 10, 0.5) + B(6; 10, 0.5) + B(9; 10, 0.5) \simeq 0.33$
- Monte-Carlo simulation:
  1. Given a random variable $y \sim \mathcal{U}(0, 1)$, define "head" if $y < 0.5$, "tail" otherwise
  2. Draw 10 random variables $x_i \sim \mathcal{U}(0, 1)$, $i = 1, \ldots, 10$
  3. Count the number of heads H, and increment T if $H = 3, 6$, or 9
  4. Repeat 2.–3. N times, with N reasonably large
  5. The probability is approximately T/N
- Note that this is an integration on a probability distribution, even if it is discrete!

Example: error estimation

What is the uncertainty on the mean?

Assuming N random variables $x_i \sim \mathcal{N}(0, \sigma),\ i = 1, \ldots, N$, the estimator of the mean is: $\bar{x} = N^{-1} \sum_{i=1}^{N} x_i$ and its uncertainty is:

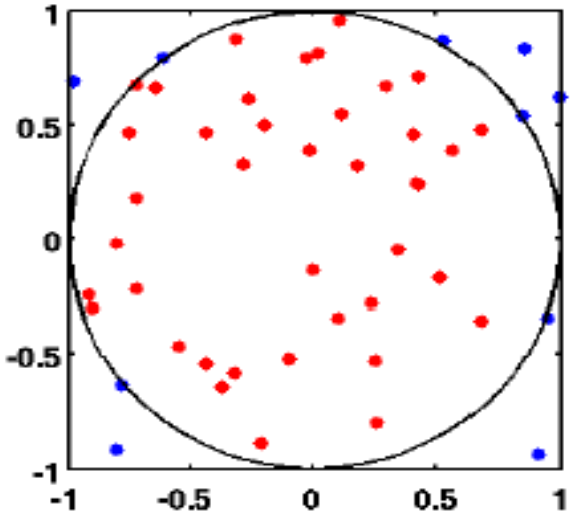
$$\sigma_{\bar{x}} = \sigma / \sqrt{N}$$

The Monte-Carlo way:

1. Draw a set of N random variables $y_i \sim \mathcal{N}(0, \sigma),\ i = 1, \ldots, N$
2. Calculate the sample mean $\bar{y} = N^{-1} \sum_{i=1}^{N} y_i$
3. Redo 1.–2. M times
4. The uncertainty on the mean $\sigma_{\bar{x}}$ is the root mean square of $\bar{y}_j,\ j = 1, \ldots, M$, i.e. $\sigma_{\bar{x}}^2 = (M-1)^{-1} \sum_{j=1}^{M} (\bar{y}_j - \hat{y})^2$, with $\hat{y} = M^{-1} \sum_{j=1}^{M} \bar{y}_j$

Example: numerical integration

How to calculate π?



1. Draw N point (x, y) uniformly at random in a square
2. Count the C points for which $x^2 + y^2 < 1$
3. The ratio C/N converges towards $\pi/4$ as $N^{1/2}$

Random numer generators

Basic principles

- We want to draw many random variables $N_i \sim \mathcal{U}(0,1)$, $i = 1, \ldots$ which satisfy (or approximate sufficiently well) all randomness properties
- $N_i \sim \mathcal{U}(0,1)$, $\forall i$ is not sufficient. We also want that $f(N_i, N_j, \ldots)$ $\forall i, j, \ldots$ has also the right properties
- Correlations in k-space are often found with a bad random-number generators
- Another issue is the period of the generator
- The `ran()` function in `libc` has been (very) bad. Do not use this function in applications when good randomness is needed says `man 3 rand`

Random numer generators

Basic algorithm

- Many random number generators are based on the recurrence relation:

$$N_{j+1} = a \cdot N_j + c \pmod{m}$$

  These are called linear congruential generators. c is actually useless.
- "Divide" by $m + 1$ to get a number in the range $[0; 1[$
- Choices of a, m in standard libraries are found to range from very bad to relatively good
- A "minimal standard" set is $a = 7^5 = 16807$, $c = 0$, $m = 2^{31} - 1 = 2147483647$. This is RAN0
- Note that the period is at most m

Random numer generators

Improvements on RAN0

1. Multiplication by a doesn't span the whole range of values, i.e. if $N_i = 10^{-6}$, $N_{i+1} \leq 0.016$, failing a simple statistical test
   - Swap consecutive output values: Generate a few values (~ 32), and at each new call pick one at random. This is RAN1
2. The period $m = 2^{31} - 1$ might be too short
   - Add the outcome of two RAN1 generators with (slightly) different m's (and a's). The period is the least common multiple of m_1 and $m_2 \sim 2 \cdot 10^{18}$. This is RAN2
3. The generator is too slow
   - Use in C inline $N_{i+1} = 1664525 \cdot N_i + 1013904223$ using `unsigned long`. Patch the bits into a real number (machine dependent). This is RANQD2

Random numer generators

Implementations and recommendations

NR: Numerical Recipes
GSL: GNU Scientific Library

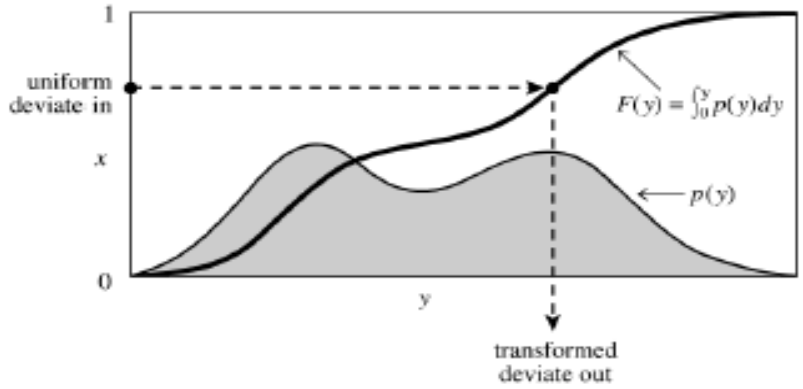
Library	Generator	Relative speed	Period
NR	RAN0	1.0	$\sim 2^{31}$
NR	RAN1	1.3	$\sim 2^{36}$
NR	RAN2	2.0	$\sim 2^{62}$
NR	RANQD2	0.25	$\sim 2^{30}$
GSL	MT19937	0.8	$\sim 2^{19937}$
GSL	TAUS	0.6	$\sim 2^{88}$
GSL	RANLXD2	8.0	$\sim 2^{400}$

Always use GSL! See the GSL doc for the many more algorithms available

Transformation method

The method

The transformation method allows in principle to draw values at random from any distribution



1. Given a distribution $p(y)$, the cumulative distribution function (CDF) of $p(y)$ is $F(y) = \int_0^y p(w)\,\mathrm{d}w$
2. We want to draw y uniformly in the shaded area, i.e. uniformly over $F(y)$; by construction $0 \leq F(y) \leq 1$,
3. We draw $x \sim \mathcal{U}(0,1)$ and find y so that $x = F(y)$
4. Therefore $y(x) = F^{-1}(x),\ x \sim \mathcal{U}(0,1)$

Transformation method

Example

Exponential deviates: $p(y) = \lambda e^{-\lambda y}$

$$F(y) = 1 - e^{-\lambda y} = x$$

$$y(x) = -\frac{1}{\lambda}\ln(1 - x)$$

Note: this is equivalent to

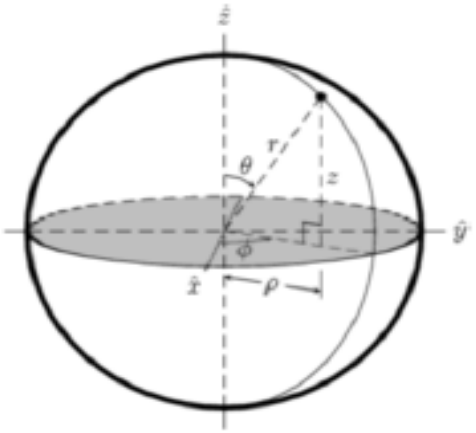
$$y(x) = -\frac{1}{\lambda}\ln(x),$$

since, if $x \sim \mathcal{U}(0, 1)$, then $1 - x \sim \mathcal{U}(0, 1)$ as well

Note also that it is rather uncommon to be able to calculate $F^{-1}(x)$ analytically. Depending on accuracy, it is possible to calculate an numerical approximation

Transformation method

A point in space



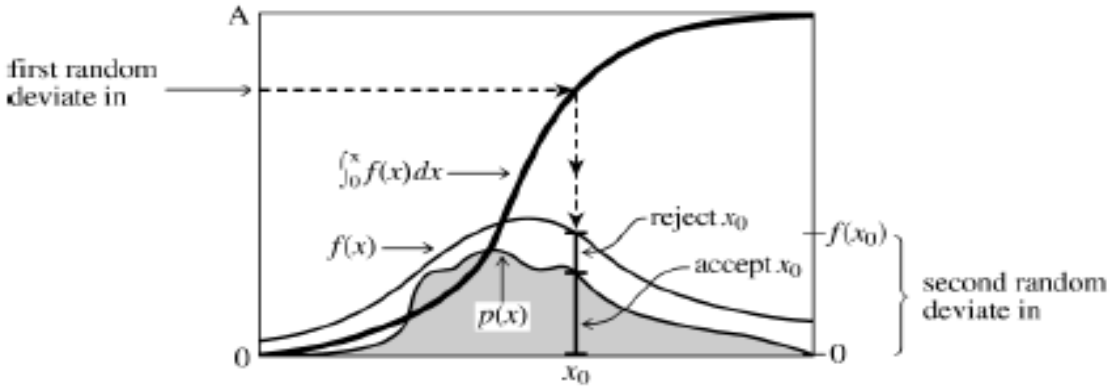
To draw a point in a homogeneous sphere of radius R:

1. ϕ can be drawn uniformly from $\mathcal{U}(0, 2\pi)$
2. θ has a sine distribution $p(\theta) = sin(\theta)/2,\ \theta \in [0; \pi[$
   Transformation: $\theta = 2\arccos(x)$
3. Each radius shell has a volume $f(R) \sim R^2\ \mathrm{d}R$, so $R \propto \sqrt[3]{x}$
4. Alternatively, draw a point at random on the surface of a sphere $(x, y, z)/\sqrt{x^2 + y^2 + z^2}$ with $x,\ y,\ z \sim \mathcal{N}(0, 1)$

Rejection method

The method

If the CDF of $p(x)$ is difficult to estimate (and you can forget about inversion), the rejection method can be used

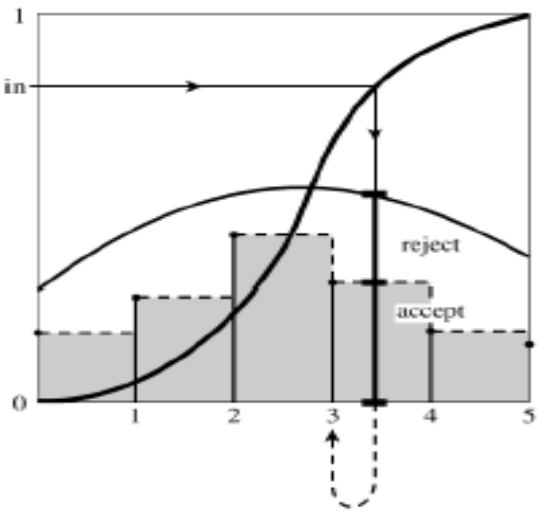


1. Find a comparison function $f(x)$ that can be sampled, so that $f(x) \geq p(x),\ \forall x$
2. Draw a random deviate x_0 from $f(x)$
3. Draw a uniform random deviate y_0 from $\mathcal{U}(0, f(x_0))$
4. If $y_0 < p(x_0)$, accept x_0, otherwise discard it
5. Repeat 2.–4. until you have enough values

The rejection method can be very inefficient if $f(x)$ is very different from $p(x)$

Rejection method

Example



The Poisson distribution is discrete: $\mathcal{P}(n;\alpha) = \frac{\alpha^n\, e^{-\alpha}}{n!}$

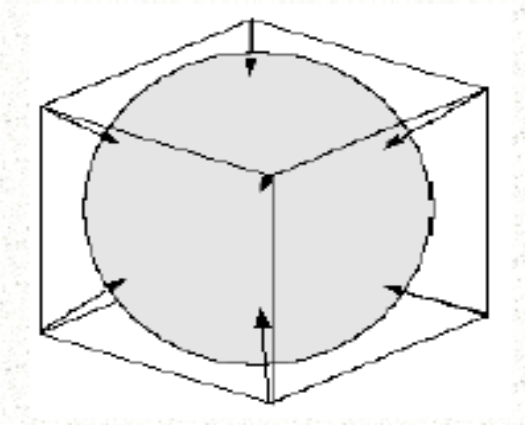
Make it continuous:

$$\mathcal{P}(x;\alpha) = \frac{\alpha^{[x]}\, e^{-\alpha}}{[x]!}$$

A Lorentzian $f(x) \propto \frac{1}{(x-\alpha)^2+c^2}$ is a good comparison function

Rejection method

A point in space



A simpler way to draw a point in a homogeneous sphere of radius R based on rejection:

1. Draw three random variables x, y, z from $\mathcal{U}(-R, R)$
2. Keep if $x^2 + y^2 + z^2 < R^2$, reject otherwise
3. Repeat 1.–2. until you have enough values

Efficiency is $\frac{4\pi}{3}/2^3 \simeq 0.52$

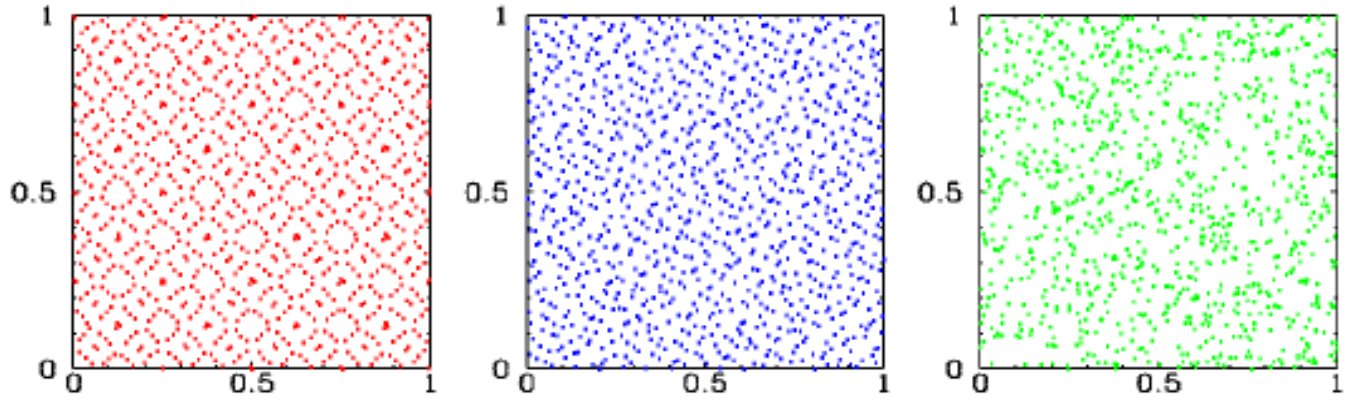
Distributions

GNU Scientific Library implements (not exhaustive!):

Gaussian	Binomial
Correlated bivariate Gaussian	Poisson
Exponential	
Laplace	
Cauchy	Spherical 2D, 3D
Rayleigh	
Landau	
Log-normal	
Gamma, beta	
χ^2, F, t	

Quasi-random numbers

What is random?



All sets of points fill "randomly" the area [[0; 1]; [0; 1]]

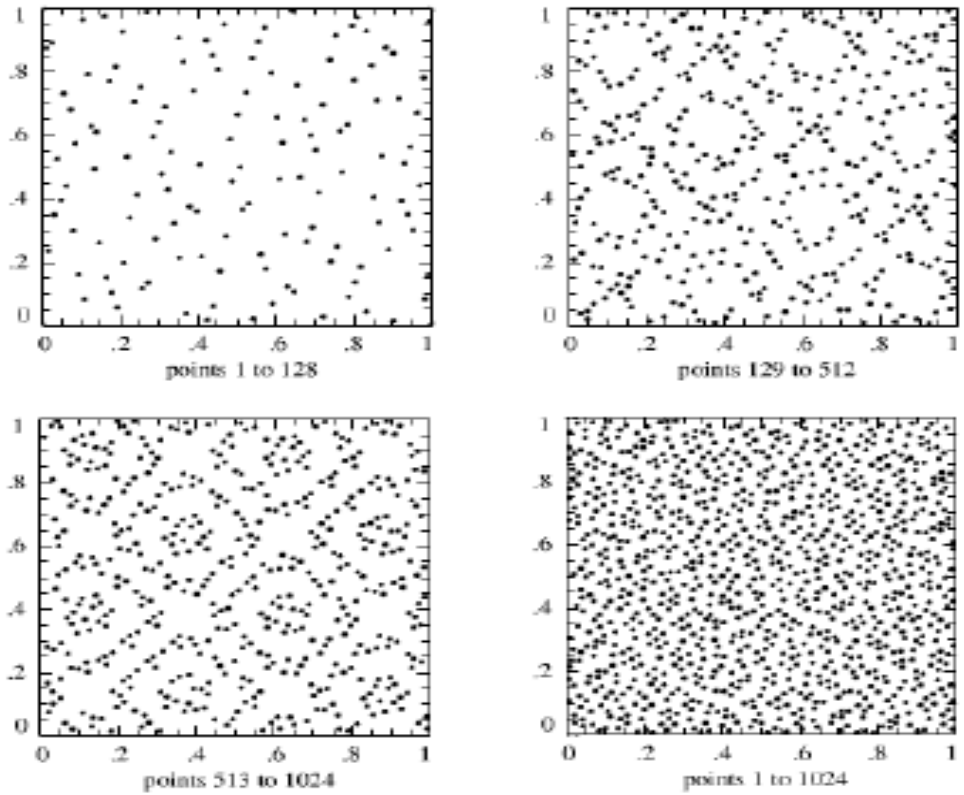
The left and center images are "sub-random" and fill more uniformly the area

These sequences are also called low-discrepancy sequences

These sequences can be used to replace the RNG when $x \sim \mathcal{U}(a, b)$ is needed

Quasi-random numbers

Filling of the plane



The sequence fills more or less uniformly the plane $\forall N$

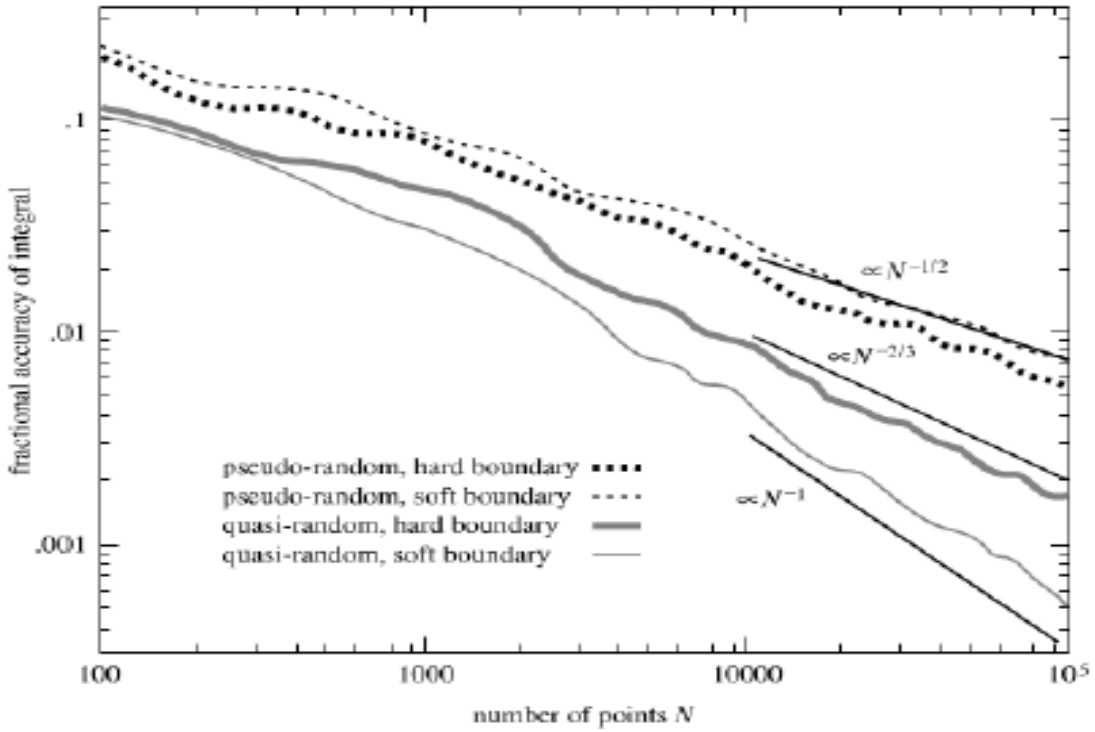
Quasi-random numbers

Examples of algorithms

- Sobol's sequence: Count in binary, but using Gray code and put a radix in front: 0.1, 0.11, 0.01, 0.011, 0.001, 0.101, 0.111, ...
  This can be generalized to N dimensions
  This is the red set of points
- Halton's sequence: $H(i)$ is constructed the following way: take i expressed in a (small) prime-number base b (say $b = 3$), e.g. $i = 17 \equiv 122$ (base 3).
  Reverse the digits and put a radix in front, i.e.
  $H(17) = 0.221$ (base 3) $\simeq 0.92593$
  This is generalized to N dimensions by choosing different b's in each dimension
  This is the blue set of points

Quasi-random numbers

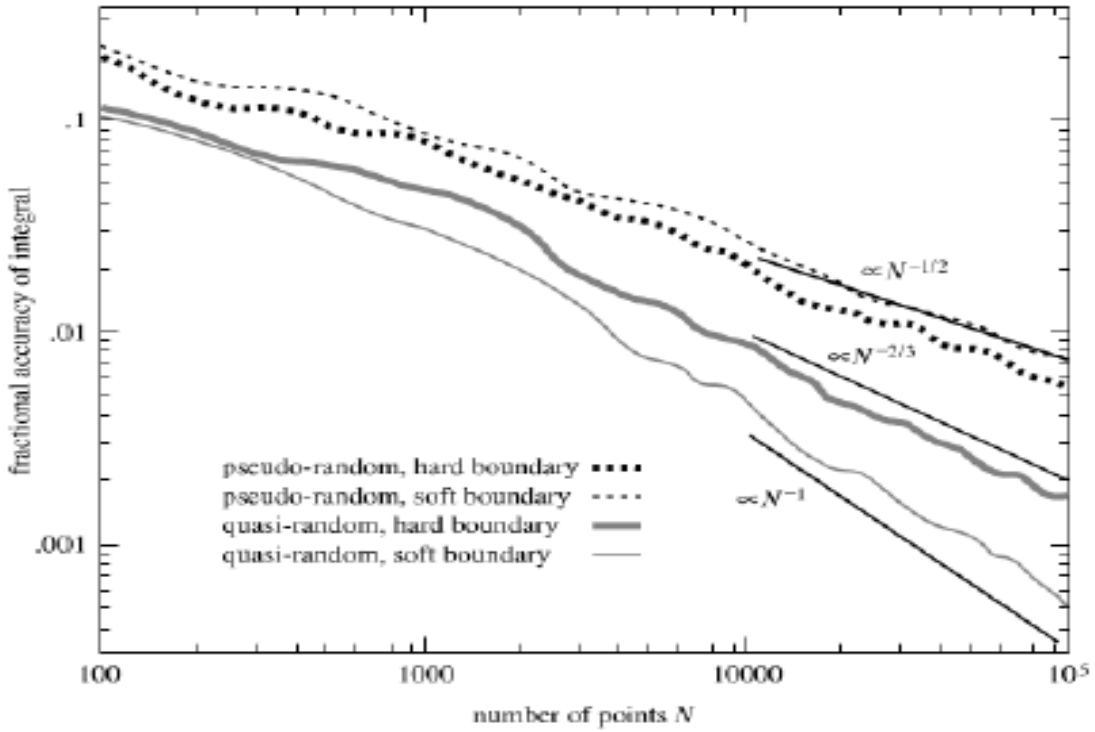
Accuracy



Convergence in some cases of numerical integration can reach $\sim 1/N$

Quasi-random numbers

Accuracy



Convergence in some cases of numerical integration can reach $\sim 1/N$

BONUS slides: For more mathematically oriented students

Follow the course/slides from
M. Chrzaszcz , Lectures on Monte Carlo methods, ETH Zurich

Monte Carlo methods: more maths

Monte Carlo method is any technique that uses *random numbers* to solve a given mathematical problem.

$\rightarrowtail$ Random number: For the purpose of this course we need to assume that we know what it is, although the formal definition is highly non-trivial.

$\Rrightarrow$ (Halton 1970): more complicated, but more accurate.

"Representing the solution of a problem as a parameter of a hypothetical population, and using a random sequence of numbers to construct a sample of the population, from which statistical estimates of the parameter can be obtained."

To put this definition in mathematical language:
Let F be a solution of a given mathematical problem. The estimate of the result $\hat{F}$:

$$\hat{F} = f(\{r_1, r_2, r_3, ..., r_n\}; ...),$$

where $\{r_1, r_2, r_3, ..., r_n\}$ are random numbers.

The problem we are solving doesn't need to be stochastic!

$\twoheadrightarrow$ One could wonder why are we trying to add all the stochastic properties to a deterministic problem. Those are the properties that allow to use all well known statistic theorems.

Applications of MC methods

⇒ Application of a MC method doesn't depend on the stochastic nature of the problem, but only on ability to represent a problem by a given hypothetical population so we can apply random numbers in that problem.

⇒ For example the calculations made in Los Alamos are so-called direct simulation. They really simulated neutron transportation in the material. The solutions are again to a non deterministic problem. Clearly 2 atomic bombs have equal energies...

⇒ The nature of the problem can be probabilistic, in which case we are performing so-called direct MC simulation. However when we are simulating not directly the problem but rather an abstract population we are using indirect MC method.

⇒ The application of a given method will depend only of mathematical structure of the problem.

Euler number determination

$\Rightarrow$ As mentioned before MC methods can be used to solve problems that **do not** have stochastic nature! All the integrals calculated in Los Alamos during the Manhattan project are nowadays solvable without any MC methods.
$\rightarrowtail$ Let's give a trivial example of solving a non stochastic problem: calculating Euler number e. We know that $e = 2.7182818....$ $\Rightarrow$ To calculate the $\hat{e}$ we will use the following algorithm:

- We generate a random number in range $(0, 1)$ (in stat. $\mathcal{U}(0, 1)$) until the number we generate is smaller then the previous one, aka we get the following sequence:

$$x_1 < x_2 < ... < x_{n-1} > x_n$$

- We store the number n. We repeat this experiment N times and calculate the arithmetic average of n. The obtained value is an statistical estimator of e:

$$\hat{e} = \frac{1}{N} \sum_{i=1}^{N} n_i \xrightarrow{N \to \infty} e.$$

$\Rightarrow$ Numerical example:

N	$\hat{e}$	$\hat{e} - e$
100	2.760000	0.041718
10000	2.725000	0.006718
1000000	2.718891	0.000609
100000000	2.718328	0.000046

Is this $\sim \sqrt{N}$?

Let's test the $\sqrt{N}$

$\Rightarrow$ In the last example we measured the Euler number using different number of pseudo-experiments.

$\rightarrowtail$ We compared the obtained value to the true and observed roughly a $\sqrt{N}$ dependence on the difference between the true value and the obtained one.

$\rightarrowtail$ Could we test this? YES! Lets put our experimentalist hat on!

$\rightarrowtail$ From the begging of studies they tooth us to get the error you need to repeat the measurements.

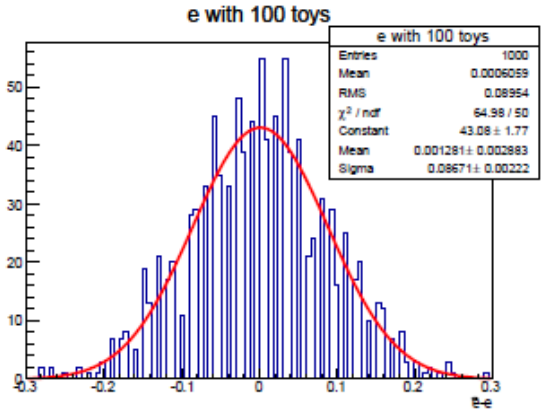
The algorithm:

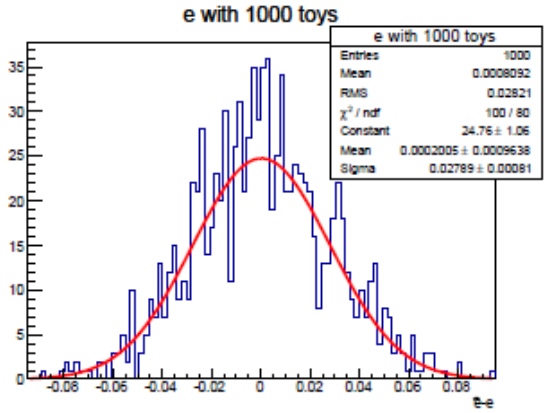
Previous time we measured Euler number using N events, where $N \in (100, 1000, 10000, 100000)$. Now lets repeat this measurement n_N times (of course each time we use new generated numbers). From the distribution of $\hat{e} - e$ we could say something about the uncertainty of our estimator for given N.

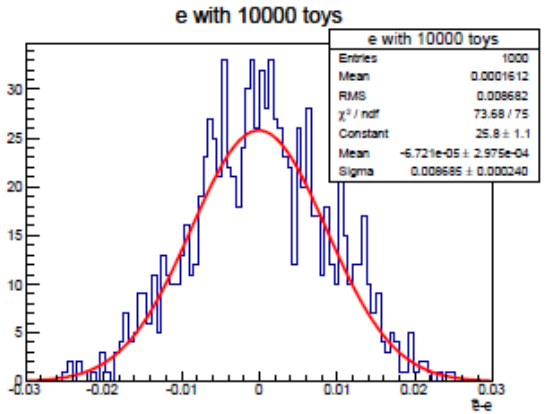
Let's test the $\sqrt{N}$

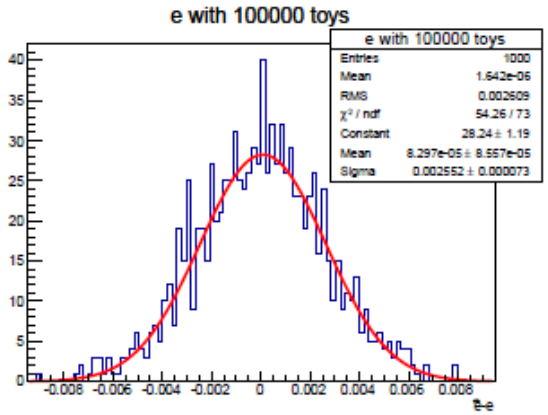
⟶ Could we test this? YES! Lets put our experimentalist hat on!

⟶ From the begging of studies they tooth us to get the error you need to repeat the measurements.





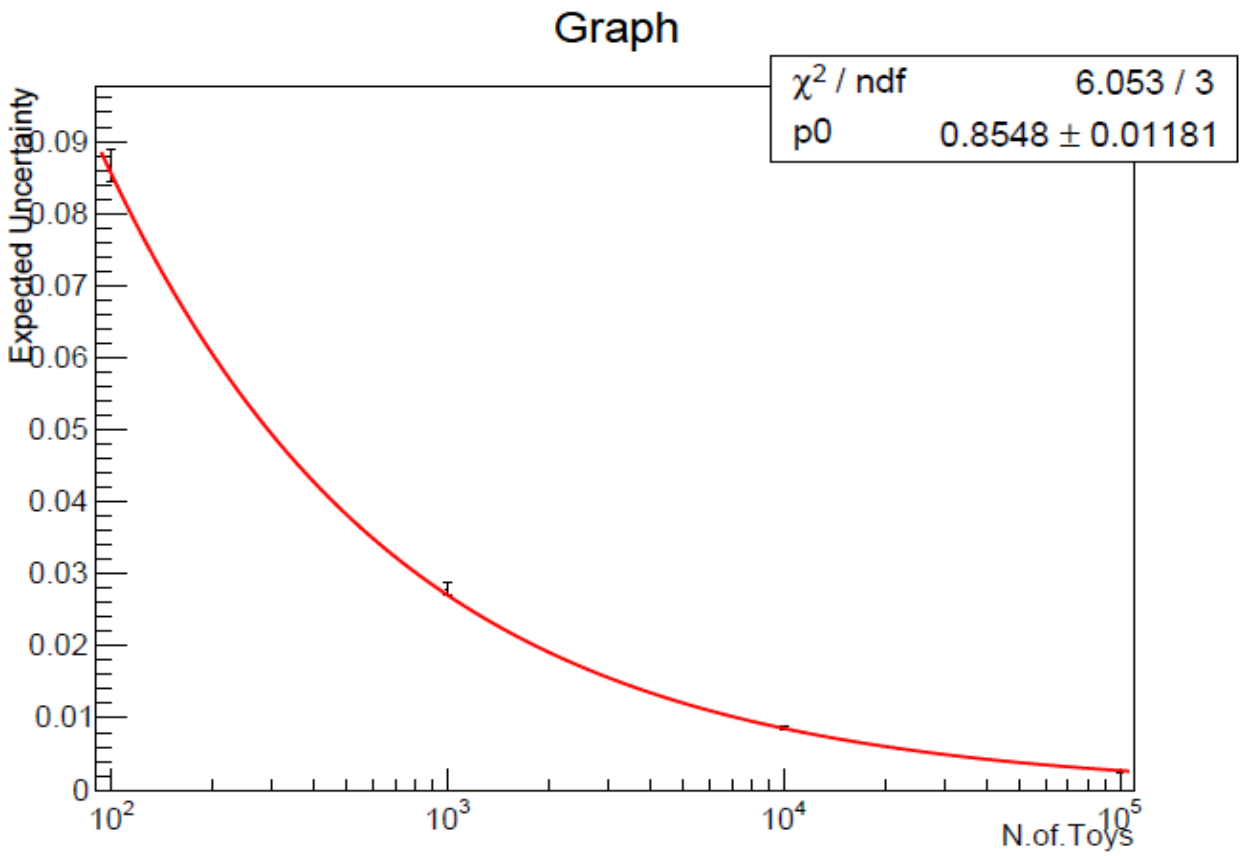




Let's test the $\sqrt{N}$

⟶ Could we test this? YES! Lets put our experimentalist hat on!

⟶ From the begging of studies they tooth us to get the error you need to repeat the measurements.



Mathematical foundations of MC methods

⇛ All MC methods are mathematically equal to a integral estimator

⇛ For simplicity lets assume that r_i are numbers from $\mathcal{U}(0,1)$.

⇛ The MC result:

$$F = F(r_1, r_2, ..., r_n)$$

is an unbias estimator of the integral:

$$I = \int_0^1 ... \int_0^1 F(x_1, ..., x_k) dx_1 ... dx_k$$

, or the expected value of F is I:

$$E(F) = I$$

⇛ This formal identity gives us the theoretical tool for application of the MC methods.

Mathematical foundations of MC methods

A random number is a number that can take more then one value (usually takes the values from continuum) and non of it's value can be predicted before hand.

$\Rightarrow$ Even thought we cannot predict the random number we can predict it's probability. $\Rightarrow$ For the continuous variables we define a *Probability Density Function* (PDF):

$$\rho(u)du = \mathcal{P}[u < u' < u + du],$$

where $\rho(u)$ is the PDF.

$\Rightarrow$ Cumulative Distribution Function (CDF):

$$R(u) = \int_{-\infty}^{u} \rho(x)dx, \qquad \rho(u) = \frac{dR(u)}{du}$$

$\Rightarrow$ The $R(u)$ is monotonically non decreasing function and takes the value in $[0, 1]$.

Mathematical foundations of MC methods

⇛ The expected value of a function $f(x)$ is defined as an average of the function

$$E(f) = \int f(u)dR(u) = \int f(u)\rho(u)du$$

⇛ If $x \in \mathcal{U}(a,b)$ then:

$$dR = \frac{du}{b-a} \qquad E(f) = \frac{1}{b-a}\int_a^b f(u)du$$

⇛ The variation is defined as:

$$V(f) = E([f - E(f)]^2) = \int [f - E(f)]^2 dR$$

⇛ The Standard deviation:

$$\sigma(f) = \sqrt{V(f)}$$

⇛ In practice we give the standard deviation not the varaition as it's the same dimension as measured quantities.

Mathematical foundations of MC methods

⇛ For 2 random variables we define:

$$E(cx+y) = cE(x) + E(y)$$
$$V(cx+y) = c^2V(x) + V(y) + 2cCov(x,y), \quad (1)$$

where $Cov(x,y) = E([x-E(x)][y-E(y)])$ is called the covariance.

$$cov(x,y) = \begin{cases} = 0 & x,y\ uncorrelated \\ > 0 & x,y\ correlated \\ < 0 & x,y\ anticorrelated \end{cases}$$

⇛ If x,y are independent then $cov(x,y) = 0$ and
$V(x+y) = V(x) + V(y)$

⇛ If variables are independent then they are uncorrelated. If they are uncorrelated then can still be dependent .

Mathematical foundations of MC methods

⇛ For 2 random variables we define:

$$E(cx+y) = cE(x) + E(y)$$
$$V(cx+y) = c^2V(x) + V(y) + 2cCov(x,y), \quad (1)$$

where $Cov(x,y) = E([x-E(x)][y-E(y)])$ is called the covariance.

$$cov(x,y) = \begin{cases} = 0 & x,y\ uncorrelated \\ > 0 & x,y\ correlated \\ < 0 & x,y\ anticorrelated \end{cases}$$

⇛ If x,y are independent then $cov(x,y) = 0$ and
$V(x+y) = V(x) + V(y)$

⇛ If variables are independent then they are uncorrelated. If they are uncorrelated then can still be dependent .

Law of large numbers

The law of large numbers (LLN): let's take n numbers from $\mathcal{U}(a, b)$ and for each of them we calculate the $f(u_i)$. The LLN:

$$\frac{1}{n}\sum_{i=1}^{n} f(u_i) \xrightarrow{N\to\infty} \frac{1}{b-a}\int_a^b f(u)du$$

$\Rrightarrow$ We say (in statistic terminology) that the left side is asymptotically equivalent to the value of the integration if $n \to \infty$.

$\Rrightarrow$ Assumptions:

- f is integrative.
- Smooth in most of the points.
- Limited.

The LLN can be interpreted as the fact the MC estimator of the integration is approaching the true value if the n is increasing.

Wrap up

⇛ From LLN:

$$\frac{1}{n}\sum_{i=1}^{n} f(u_i) \xrightarrow{N\to\infty} \frac{1}{b-a}\int_a^b f(u)du$$

⇛ Mathematical properties of MC estimator:

- If $V(f) < \infty$ then estimator is a good estimator, aka it converges to the true value.
- The estimator is an unbias estimator, aka the expected value is the true value.
- The estimator has a normal distribution for large n (CLT).
- The standard deviation of the estimator:

$$\sigma = \frac{1}{\sqrt{n}}\sqrt{V(f)}$$

Monte Carlo and integration

↪ All MC calculations are equivalent to preforming an integration.

⇉ Assumptions: r_i random numbers from $\mathcal{U}(0,1)$. The MC result:

$$F = F(r_1, r_2, ...r_n)$$

is unbias estimator of an integral:

$$I = \int_0^1 ... \int_0^1 F(x_1, x_2, ..., x_n) dx_1, dx_2 ..., dx_n$$

aka the expected value of the I integral is:

$$E(F) = I.$$

⇛ This mathematical identity is the most useful property of the MC methods. It is a link between mathematical analysis and statistic world. Now we can use the best of the both world!

If we want to calculate the integral in different range then $(0,1)$ we just scale the the previous result:

$$\frac{1}{N} \sum_{i=1}^{N} f(x_i) \xrightarrow{N \to \infty} E(f) = \frac{1}{b-a} \int_a^b f(x) dx$$

Uncertainty from MC methods

⇛ In practice we do not have $N \to \infty$ so we will never know the exact result of an integral :(

⟼ Let's use the statistical world and estimate the uncertainty of an integral in this case :)

⇀ A variance of a MC integral:

$$V(\hat{I}) = \frac{1}{n}\left\{E(f^2) - E^2(f)\right\} = \frac{1}{n}\left\{\frac{1}{b-a}\int_a^b f^2(x)dx - I^2\right\}$$

↬ To calculate $V(\hat{I})$ one needs to know the value of I!

⇛ In practice $V(\hat{I})$ is calculated via estimator:

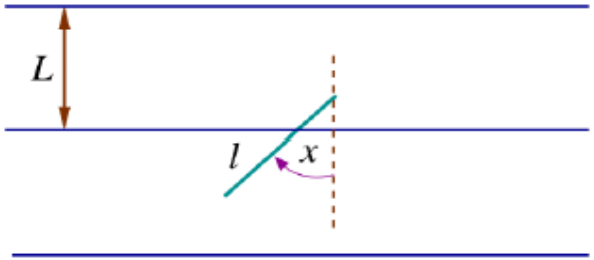
$$\hat{V}(\hat{I}) = \frac{1}{n}\hat{V}(f), \qquad \hat{V}(f) = \frac{1}{n-1}\sum_{i=1}^{n}\left[f(x_i) - \frac{1}{n}\sum_{i=1}^{n} f(x_i)\right]^2.$$

⇛ MC estimator of standard deviation: $\hat{\sigma} = \sqrt{\hat{V}(\hat{I})}$

Buffon needle – π number calculus

$\Rightarrow$ Buffon needle (Buffon 1777, Laplace 1886): We are throwing a needle (of length l) on to a surface covered with parallel lines width distance L. If a thrown needle touches a line we count a hit, else miss. Knowing the number of hits and misses one can calculate the π number.

Experiment:



n - number of hits
N number of hits and misses, aka number of tries.

Theory:

$\Rightarrow$ x - angle between needle and horizontal line, $x \in \mathcal{U}(0, \pi)$. $\Rightarrow$ the probability density function (p.d.f.) for x:

$$\rho(x) = \frac{1}{\pi}$$

$\Rightarrow$ $p(x)$ probability to hit a line for a given x value:

$$p(x) = \frac{l}{L}|\cos x|$$

$\Rightarrow$ Total hit probability:

$$P = E[p(x)] = \int_0^{\pi} p(x)\rho(x)dx = \frac{2l}{\pi L}$$

Now one can calculate $\hat{P}$ from MC : $\hat{P} = \frac{n}{N} \xrightarrow{N\to\infty} P = \frac{2l}{\pi L} \Rightarrow \hat{\pi} = \frac{2Nl}{nL}$

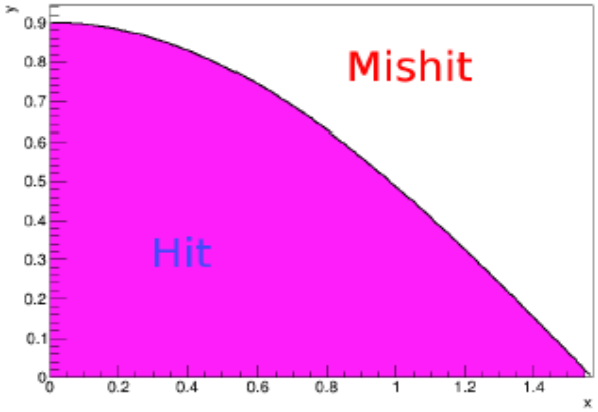
Buffon needle – simples Carlo method

Monte Carlo type "hit or miss"
Let's use the summery of $p(x)$ function and stake $0 < x < \frac{\pi}{2}$.
$\Rightarrow$ Algorithm:

Generate 2 dim. distribution:

$$(x, y) : \mathcal{U}(0, \frac{\pi}{2}) \times \mathcal{U}(0, 1) \text{ and}$$

$$y \begin{cases} \leqslant p(x) : & \text{hit,} \\ > p(x) : & \text{miss.} \end{cases}$$



Let's define weight function: $w(x, y) = \Theta(p(x) - y)$,
where $\Theta(x)$ is the step function.
$\rightarrowtail$ p.d.f.: $\varrho(x, y) = \rho(x)g(y) = \frac{2}{\pi} \cdot 1$
$\Rightarrow$ Integrated probability:

$$P = E(w) = \int w(x, y)\varrho(x, y)dxdy = \frac{2l}{\pi L} \xleftarrow{N \to \infty} \hat{P} = \frac{1}{N} \sum_{i=1}^{N} w(x_i, y_i) = \frac{n}{N}$$

Standard deviation for $\hat{P}$: $\hat{\sigma} = \frac{1}{\sqrt{N-1}} \sqrt{\frac{n}{N}\left(1 - \frac{n}{N}\right)}$

Head and tails Monte Carlo

⇛ MC estimator of an integral that is based on counting the numbers of positive trials compared to the failed ones is called "hit or miss"

⇛ The probability is described by the Bernoulli distribution:

$$\mathcal{P}(n) = \binom{N}{n} P^n (1-P)^{N-n},$$

where P is the probability of success, N is the number of trials and n is the number of successes.

⇛ The following are true:

$$E(n) = NP,$$

$$V(n) = NP(1-P),$$

⇛ Translating this into probability basis:

$$E(\hat{P}) = P, \quad V(\hat{P}) = \frac{P(1-P)}{N}.$$

Buffon needle

$\Rightarrow$ Lets make this toy experiment and calculate the π number.
$\hookrightarrow$ We can simulate the central position (y) of an needle between $(-L, L)$ from $\mathcal{U}(-L, L)$.

Symmetry:

Please note the symmetry of the problem, if the position of the needle would be $> L$ then we can shift the needle by any number of L's.

$\hookrightarrow$ New we simulate the angle (ϕ) with a flat distribution from $(0, \pi)$. $\hookrightarrow$ The maximum and minimum y position of the needle are:

$$y_{\max} = y + |\cos\phi| l$$
$$y_{\min} = y - |\cos\phi| l$$

$\hookrightarrow$ Now we check if the needle touches any of the lines: $y = L$, $y = 0$ or $y = -L$. If yes we count the events.

N	$\hat{\pi}$	$\hat{\pi} - \pi$	$\sigma(\hat{\pi})$
10000	3.12317	−0.01842	0.03047
100000	3.14707	0.00547	0.00979
1000000	3.13682	−0.00477	0.00307
10000000	3.14096	−0.00063	0.00097

Crude Monte Carlo method of integration

⇛ Crude Monte Carlo method of integration is based on Law of Large Numbers:

$$\frac{1}{N}\sum_{i=1}^{N} f(x_i) \xrightarrow{N\to\infty} \frac{1}{b-a}\int_a^b f(x)dx = E(f)$$

⇛ The standard deviation can be calculated:

$$\sigma = \frac{1}{\sqrt{N}}\sqrt{\left[E(f^2) - E^2(f)\right]}$$

⇛ From LNT we have:

$$P = \int w(x)\rho(x)dx = \int_0^{\pi/2} (\frac{l}{L}\cos x)\frac{2}{\pi}dx = \frac{2l}{\pi L} \xrightarrow{N\to\infty} \frac{1}{N}\sum_{i=1}^{N} w(x_i)$$

⇛ Important comparison between "Hit and mishit" and Crude MC methods. One can analytically calculate:

$$\hat{\sigma}^{\text{Crude}} < \hat{\sigma}^{\text{Hit and mishit}}$$

⇛ Crude MC is **always** better then "Hit and mishit" method. We will prove this on an example (can be proven analytically as well).

Crude vs „hits or miss”

⇛ The Crude MC is never worse then the "hit or miss" method.
⇛ Prove: Let's assume we calculate an integral:

$$I = \int_0^1 f(x)dx, \text{ and } 0 \leqslant f(x) \leqslant 1 \; \forall x \in (0,1)$$

⇛ The variation for the "hit-or-miss"(HM) method: $V(\hat{I}_{HM}) = \frac{1}{N}(I - I^2)$ ⇛ The variation for the crude method: $V(\hat{I}_{Crude}) = \frac{1}{N}[\int_0^1 f^2(x)dx - I^2]$ ⇛ Now the difference:

$$V(\hat{I}_{HM}) - V(\hat{I}_{Crude}) = \frac{1}{N}[I - \int_0^1 f^2(x)dx] = \frac{1}{N}\int_0^1 f(x)[1 - f(x)]dx \geqslant 0 \text{ q.e.d}$$

⇛ E2.3 Calculate the following integrals with uncertainties using "hit or miss" and crude methods:

$$\int_0^1 dx \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$$

$$\int_{x^2+y^2 \leqslant 1} \frac{1}{4}\sqrt{1 - (x^2 + y^2)}dxdy$$

Generalization to multi-dimension case. Crude method

⇛ Let $x = (x_1, x_2, ..., x_n)$- vector in the n-dim vector space $\mathcal{R}^n$.
$\Omega \subset \mathcal{R}^n$ - some subspace in the n-dim space.
$V \equiv (\Omega)$ - volume of the Ω subspaces.

$$I = \int_\Omega f(x)dx = V \int_\Omega f(x)dx/V = V \int_\Omega f(x)dp(x) \equiv VJ = VE(f),$$

where the MC estimator:

$$\hat{J} = \frac{1}{N} \sum_{i=1}^{N} f(x^{(i)}),\ x \in \mathcal{U}(\Omega)$$

⇛ The standard deviation:

$$\hat{\sigma}(\hat{J}) \frac{1}{\sqrt{N(N-1)}} \sqrt{\sum_{i=1}^{N} f^2(x^{(i)}) - \frac{1}{N}[\sum_{i=1}^{N} f(x^{(i)})]^2}$$

⇛ In the end we get:

$$\hat{I} = V\hat{J},\ \ \hat{\sigma}(\hat{I}) = V\sigma(\hat{J})$$

Generalization to multi-dimension case. „Hits or miss”

$\Rightarrow$ Let $x = (x_1, x_2, ..., x_n)$- vector in the n-dim vector space $\mathcal{R}^n$.
$\Omega \subset \mathcal{R}^n$ - some subspace in the n-dim space.
$V \equiv (\Omega)$ - volume of the Ω subspaces.

$$I = \int_\Omega dx \int_0^{f_{max}} dy \Theta(f(x) - y) = V f_{max} \int_\Omega \frac{dx}{V} \int_0^{f_{max}} \frac{dy}{f_{max}} \Theta(f(x) - y)$$

where $(x, y) \in \mathcal{U}(\Omega \times [0, f_{max}]$. $\Rightarrow$ Now we define K:

$$K = \int_\Omega \frac{dx}{V} \int_0^{f_{max}} \frac{dy}{f_{max}} \Theta(f(x) - y) = E(\Theta)$$

$\Rightarrow$ We generator: $(x, y) \in \mathcal{U}(\Omega \times [0, f_{max}])$ and check:

$$y = \begin{cases} \leqslant f(x) \text{hit, weight=1} \\ > f(x) \text{hit, weight=0} \end{cases}$$

$\Rightarrow$ In the end:

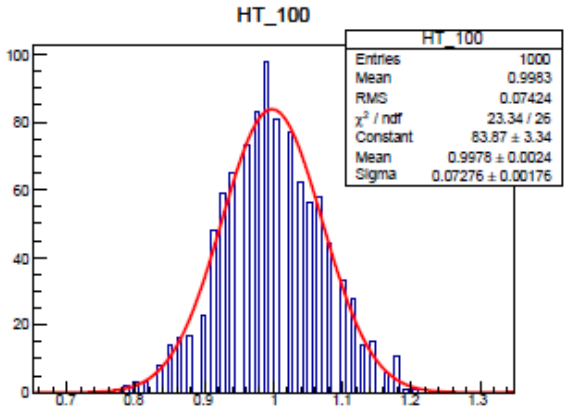
$$\hat{K} = \frac{n}{N}, \quad \hat{\sigma}(\hat{K}) = \frac{1}{\sqrt{N-1}} \sqrt{\hat{K}(1 - \hat{K})}$$

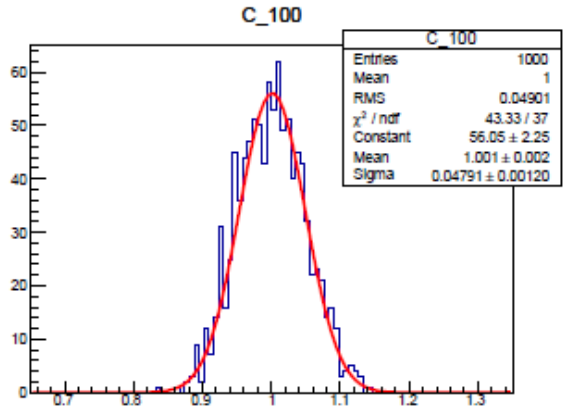
$$\hat{I} = f_{max} V \hat{K}, \quad \hat{\sigma}(\hat{I}) = f_{max} V \hat{\sigma}(\hat{K})$$

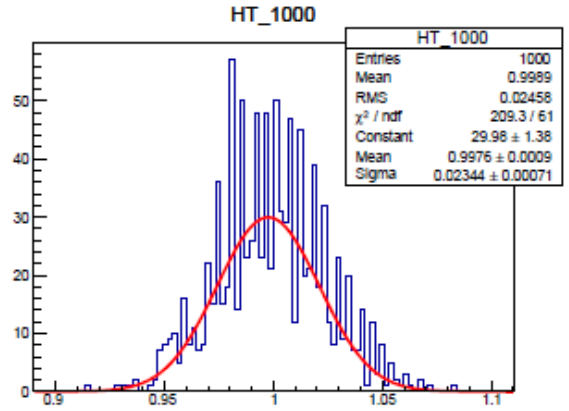
Crude MC vs „Hits or miss”

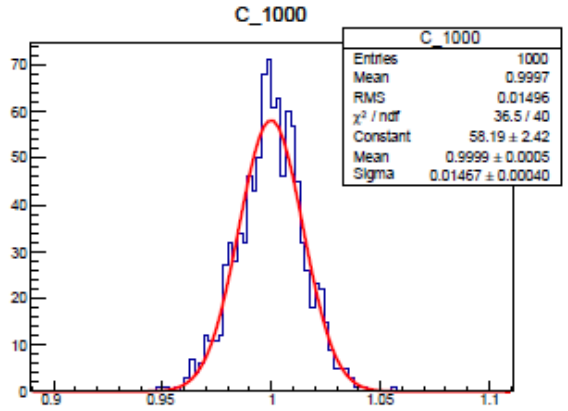
⇛ We can repeat a toy MC studies as we did in the Euler needle case.

↪ In this example we want to calculate $\int_0^{\pi/2} \cos x dx$





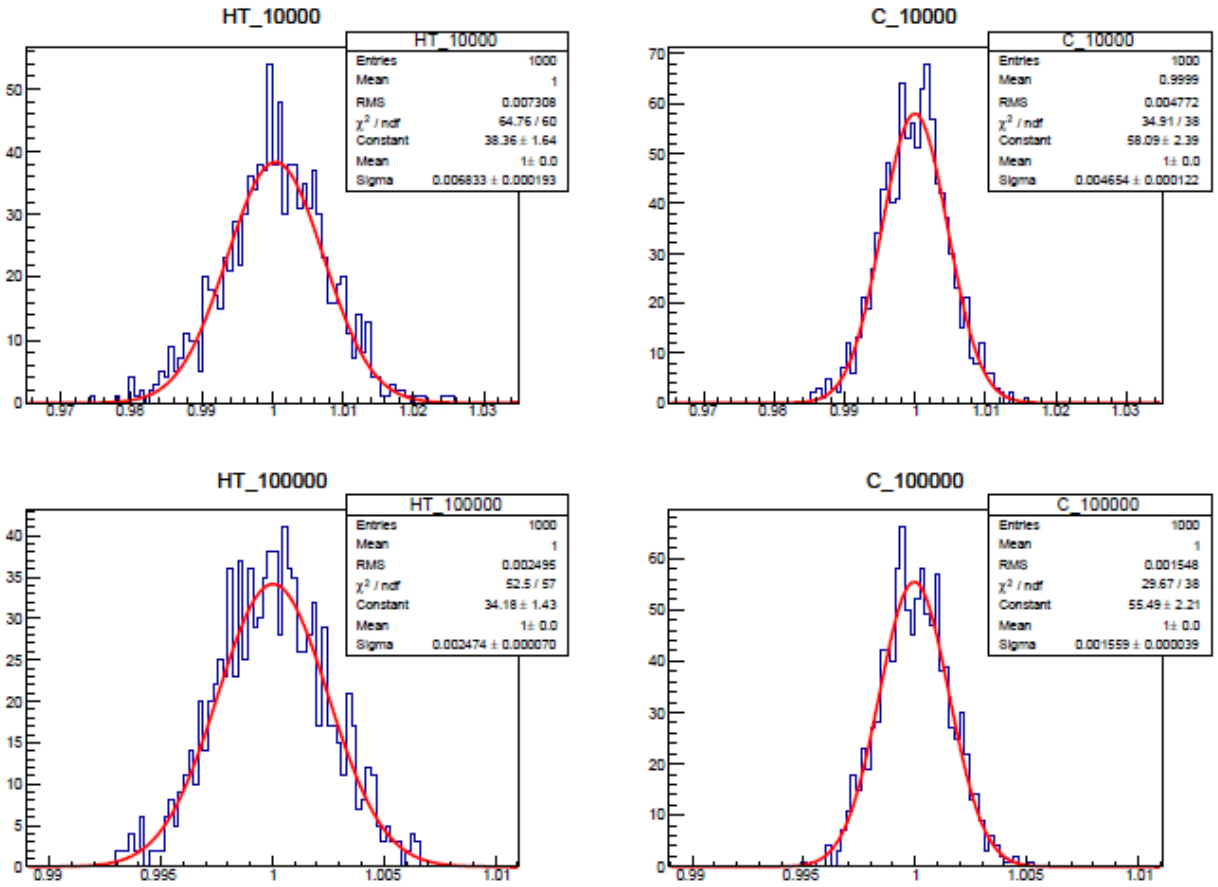




Crude MC vs „Hits or miss”

⇛ We can repeat a toy MC studies as we did in the Euler needle case.

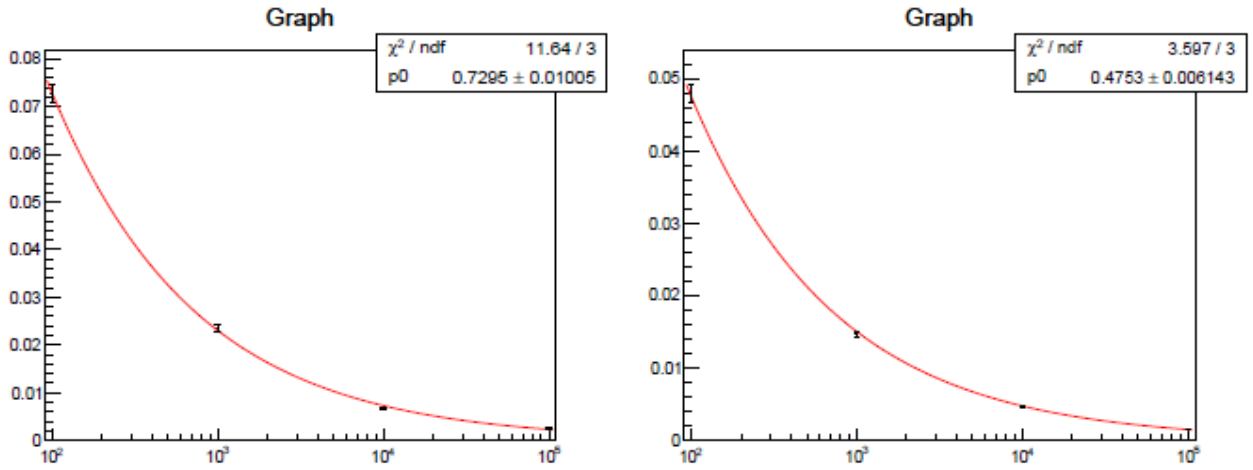
↪ In this example we want to calculate $\int_0^{\pi/2} \cos x dx$



Crude MC vs „Hits or miss”

⇛ We can repeat a toy MC studies as we did in the Euler needle case.

↪ In this example we want to calculate $\int_0^{\pi/2} \cos x dx$



⇛ One clearly sees that both methods follow $1/\sqrt{N}$ dependence and that the Crude MC is always better then the ”Hit and mishit”.

⇛ Please note that for the ”Hit and mishit” we are suing 2 times more random numbers than for the Crude method so in terms of timing the Crude MC is also much faster.

Classical methods of variance reduction

⇛ In Monte Carlo methods the statistical uncertainty is defined as:

$$\sigma = \frac{1}{\sqrt{N}} \sqrt{V(f)}$$

⇛ Obvious conclusion:

- To reduce the uncertainty one needs to increase N.
  ⇉ Slow convergence. In order to reduce the error by factor of 10 one needs to simulate factor of 100 more points!

⇛ How ever the other handle ($V(f)$) can be changed! ⟶ Lot's of theoretical effort goes into reducing this factor.

⇛ We will discuss four classical methods of variance reduction:

1. Stratified sampling.
2. Importance sampling.
3. Control variates.
4. Antithetic variates.

Stratified sampling

$\Rightarrow$ The most intuitive method of variance reduction. The idea behind it is to divide the function in different ranges and to use the Riemann integral property:

$$I = \int_0^1 f(u)du = \int_0^a f(u)du + \int_a^1 f(u)du,\ 0 < a < 1.$$

$\Rightarrow$ The reason for this method is that in smaller ranges the integration function is more flat. And it's trivial to see that the more flatter you get the smaller uncertainty.
$\rightrightarrows$ A constant function would have zero uncertainty!

General schematic:

Let's take our integration domain and divide it in smaller domains. In the j^{th} domain with the volume w_j we simulate n_j points from uniform distribution. We sum the function values in each of the simulated points for each of the domain. Finally we sum them with weights proportional to w_i and anti-proportional to n_i.

Stratified sampling – mathematical details

Let's define our integrals and domains:

$$I = \int_{\Omega} f(x)dx, \quad \Omega = \bigcup_{i=1}^{k} w_i$$

The integral over j^{th} domain:

$$I_j = \int_{w_j} f(x)dx, \quad \Rightarrow I = \sum_{j=1}^{k} I_i$$

$\rightrightarrows$ p_j uniform distribution in the w_j domain: $dp_j = \frac{dx}{w_j}$.

$\rightrightarrows$ The integral is calculated based on crude MC method. The estimator is equal:

$$\hat{I}_j = \frac{w_j}{n_j} \sum_{i=1}^{n_j} f(x_j^i)$$

Now the total integral is just a sum:

$$\hat{I} = \sum_{j=1}^{k} \hat{I}_j = \sum_{j=1}^{k} \frac{w_j}{n_j} \sum_{i=1}^{n_j} f(x_j^{(i)}),$$

Variance: $V(\hat{I}) = \sum_{j=1}^{k} \frac{w_j^2}{n_j} V_j(f)$, and it's estimator: $\hat{V}(\hat{I}) = \sum_{j=1}^{k} \frac{w_j^2}{n_j} \hat{V}_j(f)$

Stratified sampling in practice

One can show that splitting the integration region Ω into equal regions will not increase the variance!

$\Rightarrow$ For example in case of two sub samples:

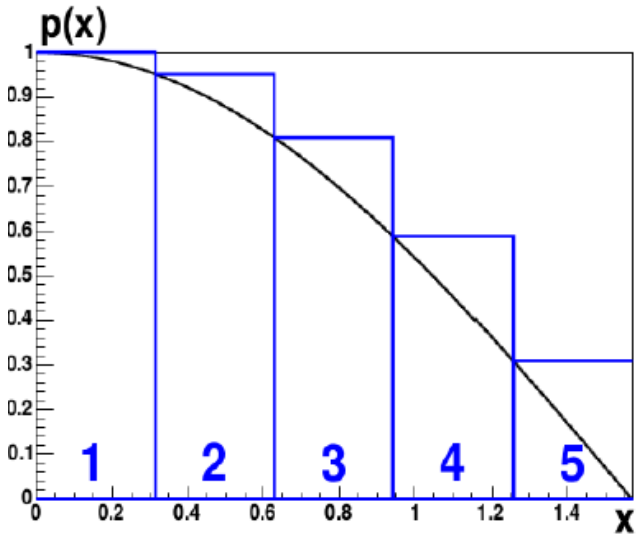
$$V(I_{\text{crude}}) - V(I_{\text{SS}}) = \frac{1}{N}\left[\int_{\omega_1} f(x)dx - \int_{\omega_2} f(x)dx\right]^{-2} \geqslant 0$$

Practical advise:

If we know very little about the integrating function the equal splitting of the Ω space is the best option!

Stratified sampling for the Buffon needle

⇛ Lets apply our Stratified sampling to my favourite :) Buffon needle with 5 samples.



⇛ We have $\omega_i = \Omega/5 = \frac{\pi}{10}$ and $n_i = \frac{N}{5}$.

⇛ The integral estimator:

$$\hat{P} = \frac{1}{\Omega} \sum_{j=1}^{5} \sum_{i=1}^{N/5} p(x_i^j) = \frac{1}{N} \sum_{i=1}^{N} p(x_i)$$

⇛ The standard deviation (for $l = L$):

$$\sigma(\hat{\pi})_{SS} = \frac{0.34}{\sqrt{N}} < \sigma(\hat{\pi})_{Crude} = \frac{1.52}{\sqrt{N}}$$

⇛ In the following example we generated a constant number of events ($N/5$) for each subsample independently of their impact on the integral.

⇛ We can improve this by generating events in each of the sub sample accordingly to the area of the blue rectangle.

⇛ E2.4 Using the Stratified Sampling please calculate the integrals from E2.3 by dividing the are into 5 samples. Compute the errors and compare them to the ones obtained from the Crude method.

Importance sampling

$\Rrightarrow$ If the function is changing rapidly in its domain one needs to use a more elegant method: make the function more stable.
$\rightrightarrows$ The solution is from first course of mathematical analysis: change the integration variable :)

$$f(x)dx \longrightarrow \frac{f(x)}{g(x)}dG(x), \text{ where } g(x) = \frac{dG(x)}{dx}$$

Schematic:

- Generate the distribution from $G(x)$ instead of $\mathcal{U}$.
- For each generate point calculate the weight: $w(x) = \frac{f(x)}{g(x)}$.
- We calculate the expected value $\hat{E}(w)$ and its variance $\hat{V}_G(w)$ for the whole sample.

- If $g(x)$ is choose correctly the resulting variance can be much smaller.
- There are some mathematical requirements:
  - $g(x)$ needs to be non-negative and analytically integrable on its domain.
  - $G(x)$ invertible or there should be a direct generator of g distribution

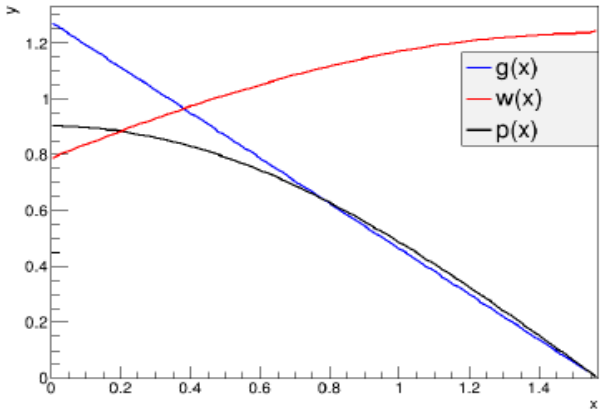
Importance sampling - Example

⇛ Let's take our good old π determination example.

⇛ Let's take here for simplicity: $L = l$.

- Let's take a trivial linear weight function: $g(x) = \frac{4}{\pi}(1 - \frac{2}{\pi}x)$
- It's invertible analytically: $G(x) = \frac{4}{\pi}x(1 - \frac{x}{\pi})$
- The weight function:

$$w(x) = \frac{p(x)}{g(x)} = \frac{\pi}{4}\frac{\cos x}{1 - 2x/\pi}$$



- Now the new standard deviation is smaller:

$$\sigma_\pi^{\mathrm{IS}} \simeq \frac{0.41}{\sqrt{N}} < \sigma_\pi \simeq \frac{1.52}{\sqrt{N}}$$

- Importance sampling has advantages:
  - Big improvements of variance reduction.
  - The only method that can cope with singularities.

⇛ Calculate the first function from E2.3 using the importance sampling. As a weight function $g(x)$ take a linear function.

Control variates

⇛ Control variates uses an other nice property of Riemann integral:

$$\int f(x)dx = \int [f(x) - g(x)]dx + \int g(x)dx$$

- $g(x)$ needs to be analytically integrable.
- The uncertainty comes only from the integral: $\int [f(x) - g(x)]dx$.
- Obviously: $V(f \to g) \xrightarrow{f \to g} 0$

⇛ Advantages:

- Quite stable, immune to the singularities.
- $g(x)$ doesn't need to be invertible analytically.

⇛ Disadvantage:

- Useful only if you know $\int g(x)dx$

Antithetic variates

⇛ In MC methods usually one uses the independent random variables. The Antithetic variates method on purpose uses a set of correlated variables (negative correlation is the important property):

- Let f and $f\prime$ will be functions of x on the same domain.
- The variance: $V(f + f\prime) = V(f) + V(f\prime) + 2Cov(f, f\prime)$.
- If $Cov(f, f\prime) < 0$ then you can reduce the variance.

⇛ Advantages:

- If you can pick up f and $f\prime$ so that they have negative correlation one can significantly reduce the variance!

⇛ Disadvantages:

- There are no general methods to produce such a negative correlations.
- Hard to generalize this for multidimensional case.
- You can't generate events from $f(x)$ with this method.

Wrap up

⇛ To sum up:

- We discussed basic mathematical properties of MC methods.
- We shown that besides the stochastic nature of MC they can be used to determine totally non stochastic quantities.
- We demonstrated there is a perfect isomorphism between MC method and integration.
- We learned how co calculate integrals and estimate the uncertainties.
- Finally we discussed several classical methods of variance reduction.

Classical methods of variance reduction

⇛ In Monte Carlo methods the statistical uncertainty is defined as:

$$\sigma = \frac{1}{\sqrt{N}} \sqrt{V(f)}$$

⇛ Obvious conclusion:

- To reduce the uncertainty one needs to increase N.
  ⇉ Slow convergence. In order to reduce the error by factor of 10 one needs to simulate factor of 100 more points!

⇛ How ever the other handle ($V(f)$) can be changed! ⟶ Lot's of theoretical effort goes into reducing this factor.

⇛ We will discuss four classical methods of variance reduction:

1. Stratified sampling.
2. Importance sampling.
3. Control variates.
4. Antithetic variates.

Disadvantages of classical variance reduction methods

⇛ All aforementioned methods(beside the Stratified sampling) require knowledge of the integration function!
⇛ If you use the method in the incorrect way, you can easily get the opposite effect than intendant.
⇛ Successful application of then require non negligible effort before running the program.
⇛ A natural solution would be that our program is "smart" enough that on his own, he will learn something about our function while he is trying to calculate the integral.
⇛ Similar techniques were already created for numerical integration!
⇛ Truly adaptive methods are nontrivial to code but are widely available in external packages as we will learn.
⇛ Naming conventions:

- Integration MC- software that is able to compute JUST! integrals.
- Generator MC- software that BESIDES! beeing able to perform the integration is also capable of performing a generation of points accordingly to the integration function.

Schematic of running this kind of methods

1. Function probing (exploration):
   - Recursive algorithm that searches for hipper-surfaces in which the function is approximately close. For evaluation of an integral in a given hipper-surface normally one uses numerical or MC crude methods. In general it is not an easy task!
   - Often the function is approximated by a given set of elementary functions.
2. Calculation phase
   - The integral is calculated using mostly using Stratified Sampling and Importance Sampling, depending on exploration phase.
   - If a MC program has capability to generated distributions accordingly to the function of which we want to calculate the integral, it's in this place where it happens.

⇛ There are algorithms where the exploration phase is linked with calculation phase. For each of the optimisation phase the integral is calculated as well. The result will be weighted average of those integrals!

This method might be bias! if in the extrapolation phase the algorithm picks up a function peaks to late the whole method will lead to systematically bias results.

VEGAS algorithm

⇛ J. G. P. Lepage (1978): adaptive algorithm for MC integration based on iterative division of the integration area (similar to `RIWID`).

⇛ Let's calculate: $\int_0^1 f(x)dx$.

- We generate M random points from $\mathcal{U}(0,1)$. We calculate from them the integral and standard deviation.
- Now we divide the integration region in N equal subdivisions:

$$0 = x_0 < x_1 < x_2 < ... < X_N = 1,\ \Delta x = x_i - x_{i-1}$$

- Now each of this subdivisions we divide further into $m_1 + 1$ subsubdivisions.

$$m_i = K \frac{\overline{f}_i \Delta x_i}{\sum_j \overline{f}_j \Delta x_j},\ K = const.\ \text{typically}\ = 1000$$

and

$$\overline{f}_i \equiv \sum_{x \in [x_{i-1}, x_i)} |f(x)| \sim \frac{1}{\Delta x_i} \int_{x_{i-1}}^{x_i} |f(x)| dx$$

⇛ The new subsubareas will be "denser" where the function is greater and less dens where the function is smaller.

VEGAS algorithm

- We are are retrieving back the original number (N) of the subdivisions by glueing together equal amount subsubdivisions.
  $\Rightarrow$ The new subdivisions will be larger where the function is larger and vice versa.
- We generate the M points accordingly to the stop function probability:

$$p(x) = \frac{1}{N\Delta x_i}$$

  and calculate the integral Stratified sampling.
- We repeat the procedure until we find an optimum division:

$$m_i \approx m_j \; i, j = 1, ..., N.$$

- In each iteration we calculate the weighted average:

$$\sum_k \frac{I_k}{\sigma_k^2},$$

  where I_k and σ_k are the integral and error in the k interaction.
- After the procedure stop we calculate the final results:

$$\hat{I} = \sigma_I^2 \sum_k \frac{I_k}{\sigma_k^2} \qquad \sigma_I = \left[\sum_k \frac{1}{\sigma_k^2}\right]^{-\frac{1}{2}}$$

VEGAS algorithm – futher improvements

⇛ In order to make the integrating area more stable(can happen that the division jumps around very rapidity). We can modify the algorithm:

$$m_i = K \left[\left[\frac{\overline{f}\Delta x_i}{\sum_j \overline{f}_j \Delta x_j} - 1 \right] \frac{1}{\log \left[\overline{f}_i \Delta x_i / \sum_j \overline{f}_j \Delta x_j \right]} \right]^{\alpha},$$

where $\alpha \in [1, 2]$ sets the convergence speed. ⇛ When function has narrow peaks the I_k and σ_k might be wrongly calculated in early stages of iteraction. To fix this we can:

$$I = \left[\sum_k \frac{I_k^2}{\sigma_k^2} \right]^{-1} \sum_k I_k \left(\frac{I_k^2}{\sigma_k^2} \right), \qquad \sigma_I = I \left[\sum_k \frac{I_k^2}{\sigma_k^2} \right]^{-0.5}$$

⇛ If the number of interactions is to large then you cannot trust the algorithm!

VEGAS algorithm – 2D case

⇛ Lets take for example $\int_0^1 dx \int_0^1 dy f(x,y)$.

⇛ We can do a trick:

$$p(x,y) = p_x(x)p_y(y)$$

⇛ One can show that using Lagrange multipliers that the optimum density has the form of:

$$p_x(x) = \frac{\sqrt{\int_0^1 dy \frac{f^2(x,y)}{p_y(y)}}}{\int_0^1 dx \sqrt{\int_0^1 dy \frac{f^2(x,y)}{p_y(y)}}}$$

⇛ So our 1D algorithm can be used to each of the axis (ex. for x axis):

$$(f_i)^2 = \sum_{x\in[x_{i-1},x_i)} \sum_y \frac{f^2(x,y)}{p_y(y)} \sim \frac{1}{\Delta x_i} \int_{x_{i-1}}^{x_i} dx \int_0^1 dy \frac{f^2(x,y)}{p_y(y)}$$

⇛ In analogous you do it for y axis.

VEGAS algorithm – an example

⇛ An example of usage: let's calculate:

$$I_n = \left(\frac{1}{a\sqrt{\pi}}\right)^n \int_0^1 \exp\left[\frac{(x_n - 0.5)^2}{a^2}\right] d^n x = 1$$

⇛ For the $n = 9$, $a = 0.1$ and $\alpha = 1$

Iteration	I_k	σ_k	I	$\sigma(I)$	Number of calculations
1	0.007	0.005	0.007	0.005	10^4
3	0.643	0.070	0.612	0.064	$3 \cdot 10^4$
5	1.009	0.041	0.963	0.034	$5 \cdot 10^4$
10	1.003	0.041	1.003	0.005	10^5
Crude MC method			0.843	0.360	10^5

FOAM algorithm

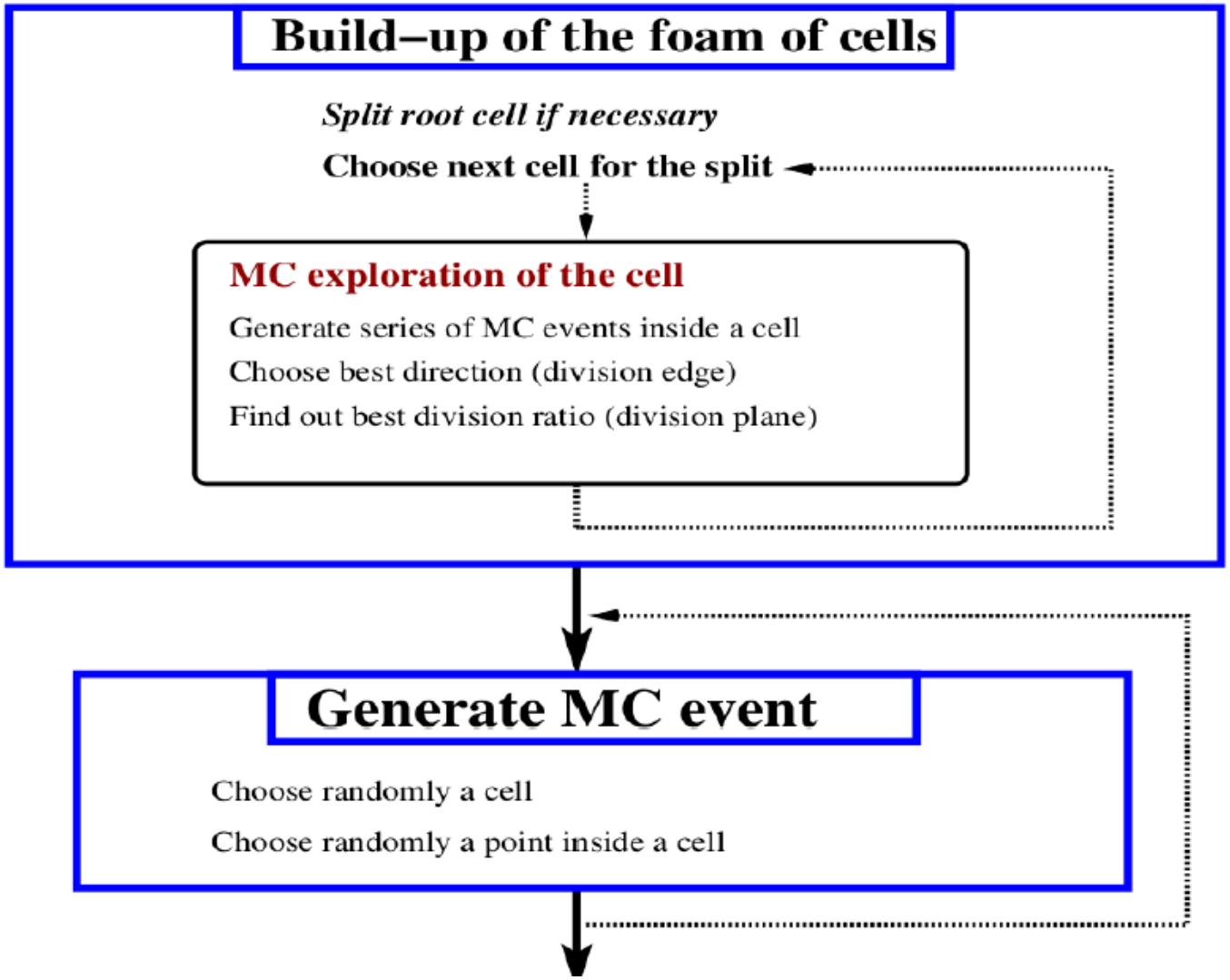
⇛ S.Jadach (2000), arXiv:physics/9910004, Comp. Phys. Commun. 152 (2003) 55.
Adaptive method with recursive division of the integration domain in cells.
⇛ There are two algorithms in dividing the integration domain:

- Symplectic: Cells are sympleces(hiper-triangles). This method can be applied to not so large number of dimensions. $(\leqslant 5)$.
- Qubic: Cells are hiper-cubes. This might be applied in higher number dimensions. $(\leqslant 20)$.

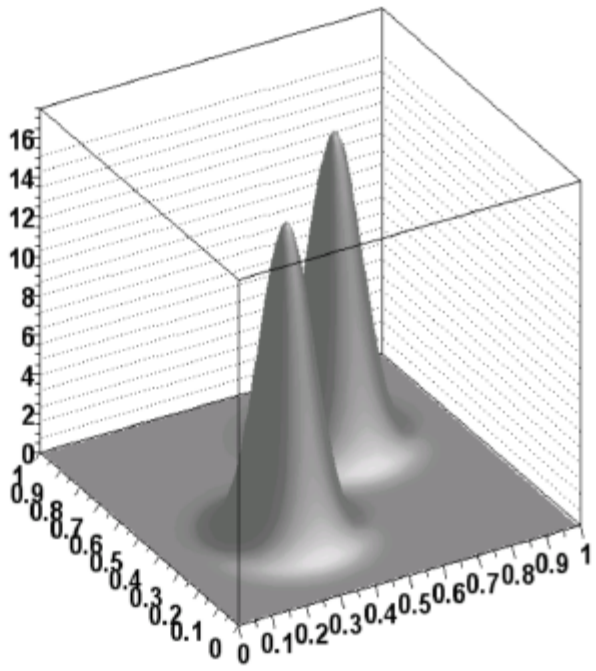
⇛ The algorithm:

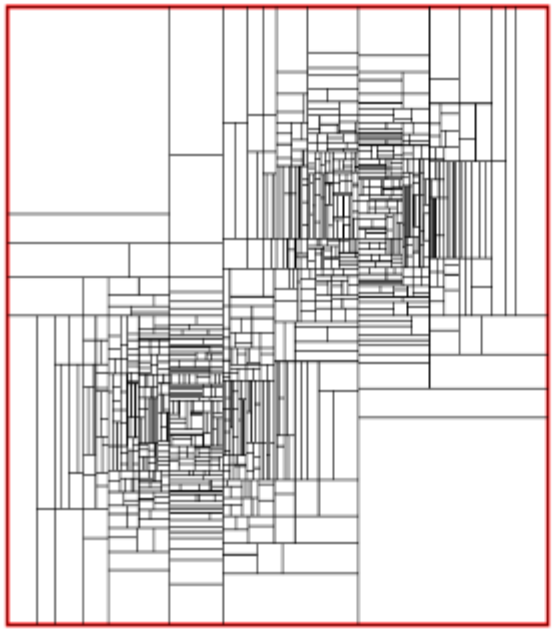
- Exploration phase:
  The integration domain (hipper-cube) is divided recursively into cells. In each step only one cell is split. The splitting is not event! The procedure is stop when the number of cells reach a certain number that is set by us. One constructs an approximation function and based on this the integral is calculated.
- Generation/Calculation Phase:
  We generate random points accordingly to the distribution of approximation function and the integral is calculated using the Importance sampling based on the approximation function.

FOAM algorithm



FOAM algorithm





Monte Carlo vs numerical methods

⇛ All numerical methods are based on evaluating the integral using linear combination of function:

$$I_Q = \sim \sum_{i=1}^{m} \omega_i f(x_i)$$

⇛ Different methods have different weights ω_i and lattice point x_i.

⇛ Efficiency of Monte Carlo methods compared to the numerical ones:

Standard deviation	1D	nD
Monte Carlo	$n^{-1/2}$	$n^{-1/2}$
Trapezoidal Rule	n^{-2}	$n^{-2/d}$
Simpson Rule	n^{-2}	$n^{-2/d}$
m-point Gauss rule	n^{-2m}	$n^{-2m/d}$

Wrap up

⇛ In one dimension the Monte Carlo method is substantially slower then the numerical methods! Even the most simple ones.
⇛ In many dimensions the Monte Carlo methods rapidity gain the advantages!
⇛ For $d > 4$ the MC method if faster then the Trapezoidal Rule.
⇛ For $d > 8$ the MC method if faster then the Simpson Rule.
⇛ The disadvantages of the numerical methods:

- Hard to apply in multi dimensions.
- Hard to apply in complex integration domains.
- The integration uncertainties are hard to evaluate.

Random and pseudorandom numbers

John von Neumann:

"Any one who considers arithmetical methods of producing random digits is, of course, in a state of sin. For, as has been pointed out several times, there is no such thing as a random number — there are only methods to produce random numbers, and a strict arithmetic procedure of course is not such a method."

⇛ Random number: a given value that is taken by a random variable
↠ by definition cannot be predicted.
⇛ Sources of truly random numbers:

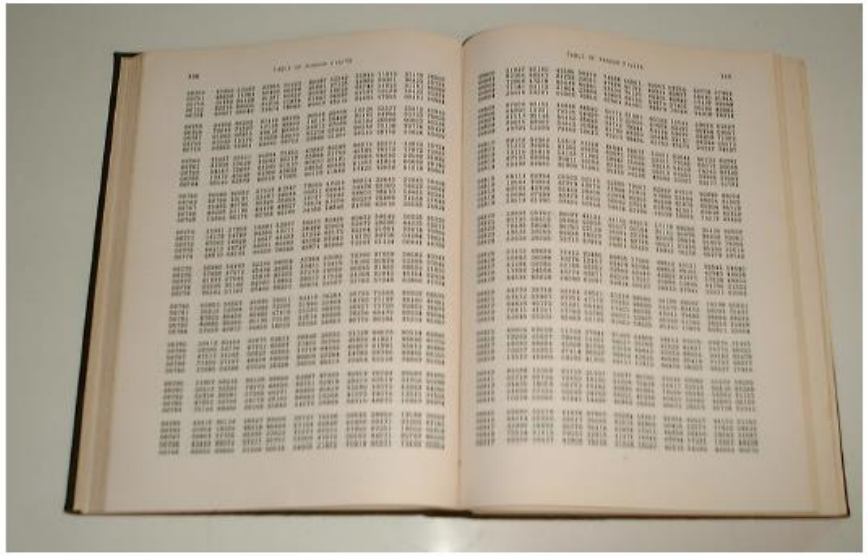
- Mechanical
- Physical

⇛ Disadvantages of physical generators:

- To slow for typical applications, especially the mechanical ones!
- Not stable; small changes in boundary conditions might lead to completely different results!

Random numbers = history remark

⇛ In the past there were books with random numbers:



⇛ It's obvious that they didn't become very popular ;)

⇛ This methods are comming back!

↠ Storage device are getting more cheap and bigger (CD, DVD).

↠ 1995: G. Marsaglia, $650\mathrm{MB}$ of random numbers, "White and Black Noise".

Pseudorandom numbers

Commercially available physical generators of random numbers are usually based on electronic noise. This kind of generators do not pass simple statistical tests! Before you use them check they statistical properties.

⇛ Pseudorandom numbers- numbers generated accordingly to strict mathematical formula.

⇛ Strictly speaking they are non random numbers, how ever they have all the statistical properties of random numbers.

⇛ How ever modern generators are so good that no one can distinguish the pseudo random numbers generated by then from true random numbers.

⇛ Mathematical methods of producing pseudorandom numbers:

- Good statistical properties of generated numbers.
- Easy to use and fast!
- Reproducible!

⇛ Because of those properties the truelly random numbers are not used in practice any more!

General schematic

⇛ Typical MC generator layout:

- We choose initial constants: X_0, X_1, ... X_{k-1}.
- The k number if calculated based on the previous ones:

$$X_k = f(X_0, ..., X_{k-1}),$$

⇛ Typically one generates $0/1$ which are then converted towards double precision numbers with $\mathcal{U}(0,1)$.

⇛ Generator period (P, l integer numbers): P is the period:

$$\exists_{l,P} : X_i = X_{i+j \cdot P} \quad \forall_{j \in \mathbb{I}+} \; \forall_{i>l}$$

⇛ In post of the cases the period can be calculated analytically, although this is sometimes not trivial.

⇛ There is a recommendation about the period of a generator. For N numbers we usually require:

$$N \ll P$$

⇛ In practice: $N < P^{2/3}$ is oki ;)

⇛ For example a generator "Mersenne Twister" (Matsumoto, Nishimura, 1998): $P \sim 10^{6000}$.

Linear generators

⇛ General equation:

$$X_n = (a_1 X_{n-1} + a_2 X_{n-2} + \ldots + a_k X_{n-k} + c) \bmod m,$$

↬ where a_i, c, m are parameters of a generator(integer numbers).

↬ Generator initialization ⇄ setting those parameters.

⇛ Very old generators. (often used in Pascal, or first C versions):

$$k = 1: \; X_n = (aX_{n-1} + c) \bmod m,$$

$$c = \begin{cases} = 0, \text{multiplicative geneator} \\ \neq 0, \text{mix geneator} \end{cases}$$

⇛ The period can be achieved by tuning the seed parameters (multiplicative) :

$$P_{\max} = \begin{cases} 2^{L-2}; \text{ for } m = 2^L \\ m - 1; \text{ for } m = \text{prime number} \end{cases}$$

Linear generators

⇛ Some simple linear generators and their periods:

a	c	m	Name/author
2^16+3	0	2^{31}	RANDU
$2^2 \cdot 23^7 + 1$	0	2^{35}	Zielinski (1966)
69069	1	2^{32}	Marsaglia (1972)
16807	0	$2^{31}-1$	Park, Miller (1980)
40692	0	$2^{31}-249$	L' Ecuyer (1988)
68909602460261	0	2^{48}	Fishman (1990)

⇛ m - prime number → better statistical properties. ⇛ There are some quid lines how to choose the parameters to make the period larger.

The periods of $2^{32} \sim 4 \cdot 10^9$ are not good enough for modern applications!
Remember that in practice $N \ll P^{2/3}$!

⇛ Simple linear generators do not pass newer statistical tests!

Linear generators

⇛ Marsaglia (1995) generators:

1. $X_n = (1176X_{n-1} + 1476X_{n-2} + 1776X_{n-3}) \bmod m,\ m = 2^{32} - 5$
2. $X_n = 2^{13}(X_{n1} + X_{n2} + X_{n3}) \bmod m,\ m = 2^{32} - 5$
3. $X_n = (1995X_{n1} + 1998X_{n2} + 2001X_{n3}) mod m,\ m = 2^{35}849$
4. $X_n = 2^{19}(X_{n1} + X_{n2} + X_{n3}) mod m,\ m = 2^{32}1629$

⇛ $P = m^3 - 1$ ⇛ They got surprisingly good statistical properties! ⇛ The main disadvantage is that multidimensional distributions look very suspicious:

$$U_i = X_i/m,\ i = 1, 2... \Rightarrow U_i(0, 1)$$

$$(U_1, U_2, ..., U_k), (U_2, U_3, ..., U_{k+1}), ...(U_1, U_2, ..., U_k), (U_{k+1}, U_{k+2}, ..., U_{2k}), ...$$

are being located on a resurfaces in a hiper-cube $[0, 1]]^k$.

⇛ Using Fourier analysis one can find the distances between the hiper-surfaces.

⇛ Generalization for multiple dimensions:

$$X_n = \mathbf{A}\vec{X}_{n-1} \bmod m,$$

RANLUX generator

⇛ All described generators are based on some mathematical algorithms and recursion. The typical scheme is of constructing a MC generator:

- Think of a formula that takes some initial values.
- Generate large number of random numbers and put them through statistical tests.
- If the test are positive we accept the the generator.

⇛ Now let's think: why the hell numbers obtained that way are showing some random number properties? There is no science behind it, it's pure luck!
⇛ M.Luscher (1993) hep-lat/9309020
⇛ Generator RANLUX based on Kolomogorow entropy and Lyapunov exponent. Effectively we are building inside the generator the chaos theory.
⇛ RANLUX and Mersenne Twister (TRandom1, TRandom3) are the 2 most powerful generators in the world that passed every known statistical test.

Wrap up

⇛ Things to remember:

- Computer cannot produce random numbers, only pseudorandom numbers.
- We use pseudorandon numbers as random numbers if they are statistically acting the same as random numbers.
- Linear generators are not commonly used nowadays.
- State of the art generators are the ones based on Kolomogorows theorem.

Literature

1. J. M. Hammersley, D. C. Hamdscomb, "Monte Carlo Methods", London: Methuen & Co. Ltd., New York: J. Wiley & Sons Inc., 1964
2. I. M. Sobol, "The Monte Carlo Method", Mir Publishers, Moscow, 1975.
3. M. H. Kalos, P. A. Whitlock, „Monte Carlo Methods", J. Wiley & Sons Inc., New York, 1986
4. G. S. Fishman, „Monte Carlo: Concepts, Algorithms and Applications", Springer, 1996.
5. R. Y. Rubinstein, D. P. Kroese, „Simulation and the Monte Carlo Method", Second Edition, J. Wiley & Sons Inc., 2008.
6. R. Korn, E. Korn, G. Kroisandt, „Monte Carlo methods and models in finance and insurance", CRC Press, Taylor & Francis Group, 2010.
7. S. Jadach, „Practical Guide to Monte Carlo", arXiv:physics/9906056, http://cern.ch/jadach/MCguide/.